SYMBOLIC STABILITY OF DELAY DIFFERENTIAL EQUATIONS

By

Victoria Averina

RECOMMENDED:

Jill Faudree

Jon Weiss

Edward L. Bush

Advisory Committee Chair

Department Head

APPROVED:

D. Woodall

Dean, College of Science, Engineering and Mathematics

Dean of Graduate School

9-19-02

Date

SYMBOLIC STABILITY OF DELAY DIFFERENTIAL EQUATIONS

A

THESIS

Presented to the Faculty
of the University of Alaska Fairbanks
in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

By

Averina Victoria, B.S.

Fairbanks, Alaska

August 2002

Abstract

We offer a new symbolic computation of stability boundaries for linear systems of time-periodic delay differential equations with the period being equal to the delay.

We construct an approximation of the "infinite-dimensional Floquet transition matrix" U using the variation of parameters method, Picard iteration, and Chebyshev approximation techniques. A *Mathematica* program approximately computes U. We show the stability boundaries of well-known examples of delay differential equations in mathematics and mechanics.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my profound gratitude to my advisor, Dr. Edward Bueler, for his guidance, support, and great efforts to explain things clearly and concisely. Since the very first day of my attendance at UAF, and especially during my thesis-writing times, he provided sound advice, great encouragement, excellent teaching, and good company.

My special thanks to Dr. Eric Butcher whose findings initiated my thesis and who has been an example of a true scientist, always willing to share his knowledge, results, and enthusiasm with fellow scientists and graduate students. Together with Dr. Bueler, Dr. Withoff, and Haitao Ma, he made my research work fun and a fruitful experience for me.

I would like to thank all the people who work in the department and who have been my teachers, mentors, and friends. I wish to thank Dr. Rybkin for recommending and then bringing me to this University. I owe a great deal of appreciation for very interesting classes to Dr. Faudree, Dr. Thomas, Dr. Lazarevic, Dr. Wiens, and Dr. Bueler.

My special acknowledgement for the invaluable teaching knowledge and experience goes to Mr. Getz, who has shaped a lot of my teaching skills and continually offered me good advice and support.

I am indebted to my fellow math graduate students: Dmitry Nicolsky, Latrice Bowman, and Tim Carlson for creating a stimulating and fun environment in which

to learn and grow. I am especially grateful to Mikhail Korotiaev and Sergey Belov who were particularly helpful mathematically.

I could never express to the fullest, my thankfulness to all of my students and especially to those who attended Math Lab. They have not only solidified my teaching and math skills but have also become great supporters and friends.

And last but not the least, my undying gratitude is toward my family and my boyfriend Adam Pike for their love and emotional support throughout difficult times and their belief in my success.

Introduction

The basics and the mathematical formulation for delay differential equations (DDEs) were developed in the 20th century. The notion of a DDE was introduced by Myshkis in 1949 as *a differential equation involving the function "x(t)" and its derivatives not only in the argument "t", but in several values of "t."*

DDEs are used to describe many phenomena in science - economics, mechanics, population dynamics, physics, medicine, chemistry, engineering, and so on. In fact, a time delay occurs naturally in just about any interaction of the real world, hence almost any model without delays is an approximation at best. In many applications it is assumed that the future state of the system is independent of the past and is determined solely by the present. Models based on this assumption involve ordinary differential equations (ODEs) and partial differential equations. However, a more realistic model of the system must include some of the past history of the system. For example, in turning, a machine process in which the tool cuts the surface formed in the previous cut, self-excited vibrations may result from the regenerative nature of the process. In order to model this regenerative effect we need to include a time-delay into a mathematical model of the process, where the delay period is the time required for one rotation of the machined part.

Explicit forms of the solutions to the mathematical models stated by DDEs can be found for linear constant coefficient cases, but introduction of time-dependent coefficients virtually always prohibits exhibiting explicit solutions. Due to this fact

relatively little is achieved in the existing theory of DDEs on qualitative analysis of the solutions such as their asymptotic behavior, stability properties, and the possibility of bifurcation on varying a parameter contained in the equations. Techniques for determining analytical stability boundaries of system of constant coefficient delay differential equations as a function of the system of parameters are well-known (see [HL], [EN]). The symbolic computation (using computer algebra systems) of stability boundaries for system of time-periodic ordinary differential equations has also been recently demonstrated (see [BS], [SB]). However, *there are no general analytical methods for systems with both time-periodic coefficients and time-delay.*

The presented work is part of an ongoing National Science Foundation project (under Grant No. 0114500) on symbolic stability and bifurcation analysis of time-periodic delay differential equations and applications to high-speed machining models[1]. The aim of this work is to develop methods for the symbolic computation of linear stability boundaries for linear system with parameters of time-periodic delay differential equations with the period being equal to the delay.

We start by investigating an existing method on the topic utilizing the variation-of-parameters formula (Chapter 1). Then we extend this method to the approximation of the infinite-dimensional delay Floquet transition matrix U using Picard iterations and Chebyshev approximation techniques (Chapter 2). We discuss merits and demerits of some stability criteria applied to U (Chapter 3). Finally, we apply our method to well-known examples of delay differential equations in mathematics and mechanics to show that the stability boundaries are approximately correct, either by comparison to analytical results or comparison to other computations (Chapter 4). In Chapters 3 and 4 we will also discuss other ideas that we believe are valuable for improving computational aspects of the method, such as using Magnus expansion instead of Picard iterations.

[1] Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

In order to test the developed technique on examples a *Mathematica* (v4.0) program was created (see Appendices A and B). The main goal of the program is to approximately compute U and find and plot the symbolic stability boundaries in the (a, b)-plane, where a and b are parameters of a DDE system. It can also plot the numerical stability boundaries in case symbolic computations are too time-consuming. In addition the program is also designed to work with a DDE system with one parameter (find the intervals of stability symbolically) or no parameters at all (plot the solution to a system on a given interval for a given initial condition, plot the graph of the spectral radius of U versus its size). Almost all names of the variables in the program are consistent with notation used in the text, which together with comments will make it easier to use and understand.

Computing and analyzing U together with the examples are the main results of this work and represent a new promising technique for symbolic computation of stability boundaries for system of time-periodic delay differential equations.

Chapter 1

The Problem of Stability

1.1 Introduction

Consider a linear system of DDEs with fixed delay $\tau > 0$,

$$\dot{x}(t) = A(t)x(t) + B(t)x(t-\tau), \tag{1.1}$$

where $x(t) \in \mathbb{R}^n$, $A(t)$ and $B(t)$ are periodic continuous $n \times n$ matrix functions of time with period $T = \tau$[1] and depend on some unknown parameters.

In contrast to ODE it is not enough to know the vector of *values of the functions at one point* in order to specify the particular solution of (1.1). It is intuitively obvious that we need to define a vector of *functions on the entire interval* of length τ in order to obtain the unique solution. The proof of this fact can be found in [HL] and other books on DDEs. The requirement of having a vector of functions as an initial condition, i.e. from an infinite-dimensional set of initial conditions, takes us to a whole new level of analysis of even simplest DDE.

There are many definitions used for stability. We will use classical definition of stability and call a system stable if and only if its response remains bounded as time

[1]The case $T = n\tau$, where n is some integer can be handled in similar way but will not be considered in this work.

goes to infinity. So the goal of this work is to answer **the stability question**: for which values of parameters the solution to (1.1) is stable?

In the next sections we will discuss analytic methods for investigating the stability of DDEs: solving the characteristic equation and seeking the analog of the Floquet transition matrix which appears in the theory of periodic linear ODEs.

1.2 Stability chart

First, let us discuss the concept of stability charts and stability boundaries. For the application problems, like vibrating in turning, we are not concerned with solving an initial value problem. Instead, we want to know for which values of the parameters (such as speed, for example) of a given DDE system the solution is *stable*. This leads us to the idea of creating a *stability chart* – plot of one parameter against another that depicts regions for which a system is stable as opposed to those regions for which it is unstable. Stability charts provide a powerful summary of the behavior of a system with periodic coefficients.

The simplest numerical approach to creating a stability chart for a DDE system:

- take a grid of points in the parameter-plane;
- set some initial condition;
- at each point of the grid solve initial value problem and determine whether the solution is stable (then plot a black point) or unstable (plot a white point).

As a result we will get a plot with regions of black points - stability regions. The finer the grid the better we can see the regions.

This approach however has some serious flaws. For one it is not proved that stability of a DDE is independent (with a probability of measure one) of the initial condition. Moreover, there is no general way of picking the most efficient (in terms of a practical application) point in the stability region, which was obtained numerically.

Therefore, we want to obtain a stability region symbolically. In order to do this we need to obtain its boundaries. We call them *stability boundaries*. If we have the symbolic expression for the stability boundaries then we can apply usual calculus tools for choosing the most efficient point in the stability region.

1.3 Characteristic equation for constant coefficient case

By analogy with ODEs one would obtain the characteristic equation for

$$\dot{x}(t) = Ax(t) + Bx(t-\tau) \tag{1.2}$$

by seeking nontrivial solutions of the form $e^{\lambda t}c$ where c is some constant $n \times 1$ vector. We have the following definitions:

Definition 1. The function $\mathcal{F} : \mathcal{C} \to \mathcal{C}$ given by:

$$\mathcal{F}(\lambda) = \det(\lambda I_n - A - Be^{-\lambda\tau}),$$

is called the *characteristic function* to the linear system (1.2), and the equation $\mathcal{F} = 0$ is called the *characteristic equation* associated to the system (1.1).

Definition 2. The characteristic function given in Definition 1 is called *exponentially stable* if and only if the following holds:

$$\{\lambda \in \mathbb{C} : Re(\lambda) \geq 0, \quad \mathcal{F}(\lambda) = 0\} = \emptyset.$$

Note that the characteristic equation is transcendental (for nonzero $B(t)$) and thus has infinitely many solutions. Therefore, it is not obvious that the solutions of (1.1) can be obtained as linear combinations of the characteristic functions. It

is not even obvious that the stability of the solutions of (1.1) is determined by the solutions of the associated characteristic equation. Both of these problems have a positive solution which can be found in [HL], [EN]. Thus we can refer to the zeros of the characteristic equation as eigenvalues of the associated system. Eigenvalues of any system of DDEs have some nice and interesting properties:

Proposition 1. *If there exists a sequence* $\{\lambda_k\}_{k\geq 1}$ *of the zeros of the characteristic equation* $\mathcal{F}(\lambda) = 0$ *such that* $|\lambda_k| \to +\infty$ *as* $k \to +\infty$, *then* $Re(\lambda_k) \to -\infty$ *as* $k \to +\infty$.

Corollary 1. *There exists a* $\beta > 0$ *such that all zeros* $\{\lambda_k\}_{k\geq 1}$ *of the characteristic equation* $\mathcal{F}(\lambda) = 0$ *satisfy* $Re(\lambda_k) < \beta$ *and there are only a finite number of solutions in any vertical strip in the complex plane.*

From the Definition 2 and the Corollary 1 it follows that:

Lemma 1. *The characteristic function* $\mathcal{F}$ *with zeros* $\{\lambda_k\}_{k\geq 1}$ *is stable if and only if* $\max_k Re\lambda_k < 0$.

Theorem 1. *If characteristic function of the linear system* (1.2) *is exponentially stable then the solution is stable for any initial condition.*

Even though it is impossible to solve the characteristic equation of the time-periodic system of DDEs analytically, there are several analytical methods available for the constant coefficient system of DDEs, such as Pontryagin criterion (see [HL]).

Let us find the stability boundaries for the following scalar constant coefficient DDE:

$$\dot{x}(t) + ax(t) = bx(t-\tau). \tag{1.3}$$

The characteristic function corresponding to (1.3) is $\mathcal{F}(\lambda) = \lambda + a - be^{-\lambda\tau}$. It has a zero root for

$$a - b = 0. \tag{1.4}$$

Now let the function have a purely imaginary root $i\omega$:

$$i\omega + a - be^{-\tau i\omega} = 0 \iff i\omega + a - b(\cos(\tau\omega) - i\sin(\tau\omega)) = 0.$$

Separating real and imaginary parts we get the following parametric boundaries:

$$a = \frac{-\omega\cos\tau\omega}{\sin\tau\omega}, \quad b = \frac{-\omega}{\sin\tau\omega}. \tag{1.5}$$

The lines defined in (1.4) and (1.5) are the boundaries of a stability region and are shown in Figure 1.1. In Chapter 4 we will also obtain the approximation of these boundaries using our method (see Figure 4.7).

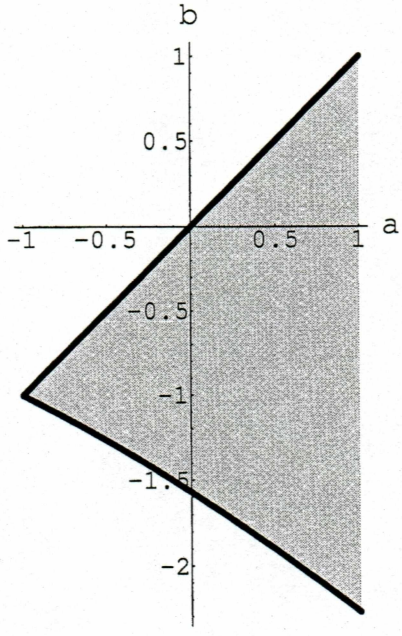

Figure 1.1: Stability boundaries (black) for $\dot{x}(t) = ax(t) + bx(t-1)$ described by (1.4) and (1.5) and stability region (shaded area).

Since the analytic conditions for the stability boundaries of a system of DDEs can be found only in the constant coefficients case, a fundamentally different approach is needed once we switch to time-periodic coefficients.

1.4 Delay Floquet Transition Matrix

Suppose together with (1.1) we are also given some initial condition $\varphi(t)$ which is $n \times 1$ vector of continuous functions of time on the interval $[-\tau, 0]$. Then if $t \in [0, \tau]$ we can treat $\varphi(t)$ as the nonhomogeneous part of a system of ODEs. Then a standard variation-of-parameters formula to construct the solution $x(t)$ on $[0, \tau]$ can be used (see [HL] or [BBA] for the proof). Obviously, we can apply the same procedure on the next intervals of length τ: $[\tau, 2\tau]$, $[2\tau, 3\tau]$, and so on. Such procedure is known as the *method of steps*. For the general case, if we know the solution on $[t_0 - \tau, t_0]$ then the solution on $[t_0, t_0 + \tau]$ can be found using the following formula:

$$x(t) = \Phi(t)\Psi(t_0)^\top x(t_0) + \int_{t_0}^{t} \Phi(t)\Psi(s)^\top B(s)x(s-\tau)ds, \tag{1.6}$$

where $\Phi(t)$ is the fundamental solution to the ODE system

$$\dot{\Phi}(t) = A(t)\Phi(t), \ \Phi(0) = I_n \tag{1.7}$$

and $\Psi(t)$ is the solution to the adjoint ODE system

$$\dot{\Psi}(t) = -A(t)^\top \Psi(t), \ \Psi(0) = I_n. \tag{1.8}$$

I_n is the identity on $\mathbb{R}^n$.

For stability analysis we need to see the solution to (1.1) as a linear map from a space of functions back to the same space of functions. This will allow us to use eigenvalue theory to determine the stability of that map.

The periodicity of matrices $A(t)$ and $B(t)$ and the fact that their period $T = \tau$ allow us to make next step and rewrite (1.6) as follows (see [BBA] for details):

$$(Ux)(t) = \Phi(t)[x(t) + \int_{0}^{t} \Psi(s)^\top B(s)x(s)ds], \quad t \in [0, \tau]. \tag{1.9}$$

This defines a linear map $U : \mathcal{C}[0, \tau] \to \mathcal{C}[0, \tau]$, where $\mathcal{C}[0, \tau]$ is a vector space of continuous $\mathbb{R}^n$-valued functions on $[0, \tau]$.

Note that (1.9) becomes a well-known formula of the solution to an ODE system once $B(t) \equiv 0$. In fact, $(U\cdot)(T)$ becomes a Floquet transition matrix $\Phi(T)$. This justifies the name for U – the *delay Floquet transition matrix* (DFTM). Moreover, $(U\cdot)(t)$ is a fundamental solution matrix for the DDE system.

Note also that the same procedure can be applied for the neutral equation:

$$\dot{x}(t) = A(t)x(t) + B(t)x(t-\tau) + C(t)\dot{x}(t-\tau),$$

where C(t) is periodic continuous $n \times n$ matrix function of time. On each interval of the length τ the delayed parts are known from the previous interval and therefore we can treat them as nonhomogeneity of the corresponding ODE.

We can view the method of steps as an iterative application of U. First let $f(t) = \varphi(t)$ for $t \in [0, \tau]$. Then:

$$\begin{aligned} x(t) &= (Uf)(t) \text{ for } t \in [0, \tau], \\ x(t) &= (U^2 f)(t-\tau) \text{ for } t \in [\tau, 2\tau], \\ x(t) &= (U^3 f)(t-\tau) \text{ for } t \in [2\tau, 3\tau], \end{aligned}$$

and so on.

Having defined U we can now answer the stability question: U (and thus the solution to (1.1)) is stable if and only if all of its eigenvalues are inside of the unit circle.

Chapter 2

Computing Delay Floquet Transition Matrix

2.1 Introduction

In the previous chapter we have shown how to derive a linear map U from the variation of parameters formula using periodicity of $A(t)$ and $B(t)$ and the fact that $T = \tau$.

Since U maps a space of functions $C[0, \tau]$ to itself it is an infinite-dimensional operator. Also U depends on the parameters represented in (1.1). For nonconstant problems one should not expect U (and as a matter of fact $\Phi(t)$ and $\Psi(t)$) to have an analytic expression. Therefore, we will need to approximate these operators and what is more approximate them uniformly, i.e. each entry should have a good approximation at any time.

Approximation of functions can be done in many different bases such as trigonometric functions, rational functions, and polynomials. It is a fact that continuous functions can be uniformly approximated by polynomials arbitrarily well (Weierstrass Approximation Theorem). Approximation with polynomials is also efficient (time-wise) in symbolic computations. In the space of polynomials the basis of Chebyshev

polynomials is known for giving very good uniform approximation of a continuous function on an interval of finite length. Among the polynomials with a good uniform approximation Chebyshev polynomials are also very easy to construct.

One of the methods for finding $\Phi(t)$ and $\Psi(t)$ requires solving a matrix inverse problem. Since symbolically that is virtually impossible to do, we will use approximation by Picard iterations. There is another method called Magnus expansion (see [M] and [I]) which we believe can improve the approximation of $\Phi(t)$ and $\Psi(t)$.

In any case, by choosing a finite set of l orthogonal polynomials for approximation we get a finite-dimensional subspace $V_l \subset \mathcal{C}[0,\tau]$ of dimension nl. Therefore, U is approximated by an $nl \times nl$ matrix $W : V_l \to V_l$.

In this chapter we will discuss the approximation by Picard iterations and Chebyshev polynomials and we will obtain expansion formula for $(Ux)(t)$.

2.2 Approximation by Picard iterations

Commonly Picard iterations are used in proofs of existence and uniqueness of solutions of ODE initial value problem. Here we will use them to find approximations for $\Phi(t)$ and $\Psi(t)$.

In order to find the solution to the ODE system (1.7) we need to solve

$$\Phi(t) = I_n + \int_0^t A(s)\Phi(s)ds. \tag{2.1}$$

Note that if $A(t)$ is a constant, or more generally commutative, matrix then $\Phi(t) = e^{\int_0^t A(s)ds}$. Recall that a matrix function $A(t)$ is commutative if and only if $A(t)A(s) = A(s)A(t)$, for any t, s.

Let us introduce the following operator:

Definition 3. If F(t) is a continuous $n \times n$ matrix-valued function then let

$$(G_A F)(t) = \int_0^t A(s)F(s)ds.$$

Now we can rewrite (2.1) as

$$\Phi(t) = I_n + (G_A\Phi)(t).$$

Therefore

$$\Phi(t) = ((1 - G_A)^{-1} I_n)(t),$$

where 1 is the identity operator on the space of $n \times n$ matrix-valued functions.

The inverse $(1-G_A)^{-1}$ can be calculated by the usual power series $1+G_A+G_A^2+....$ At the level of the matrix-valued functions this is Picard iterations. So we have the following approximation for $\Phi(t)$

$$\Phi^{(p)}(t) = ((1 + G_A + G_A^2 + G_A^3 + ... + G_A^p)I_n)(t), \tag{2.2}$$

where p is the number of Picard iterations.

Similarly,

$$\Psi^{(p)}(t) = ((1 - G_{A^T} + G_{A^T}^2 - G_{A^T}^3 + ... + (-1)^p G_{A^T}^p)I_n)(t) \tag{2.3}$$

is the Picard approximation for the adjoint equation (1.8)

The following lemma (see [BBA] for proof) gives us the formula for computing p sufficient to achieve a certain accuracy in approximating $\Phi(t)$ and $\Psi(t)$ by $\Phi^{(p)}(t)$ and $\Psi^{(p)}(t)$ correspondingly.

Lemma 2. *Let $\|\cdot\|_r$ be a matrix norm for $1 \le r \le \infty$. For t in any finite interval $[0,1]$, $\Phi^{(p)}(t) \to \Phi(t)$ as $p \to \infty$. In fact,*

$$\|\Phi(t) - \Phi^{(p)}(t)\|_r \le \sum_{q=p+1}^{\infty} \frac{\overline{a}_r^q}{q!} \tag{2.4}$$

where

$$\overline{a}_r = \max_{0 \le s \le 1} \|A(s)\|_r.$$

In practice (see Chapter 4) using eigenvalues of $A(t)$ instead of the norm yields a smaller estimate for p. However, we cannot offer a proof of this fact at the moment. We will state it as following:

Asymptotically True Lemma. *For sufficiently large p, $\overline{a}_r$ in the Lemma 2 can be replaced by*

$$\max_{0 \le s \le 1} \rho_A(s),$$

where $\rho_A(s)$ is the spectral radius (maximum in magnitude eigenvalue) of $A(s)$.

2.3 Chebyshev approximation of a scalar function

Now let us discuss how to approximate each entry of $\Phi(t)$ and $\Psi(t)$ by shifted Chebyshev polynomials. In particular we consider how to replace a scalar function by (parameter-dependent) matrix of Chebyshev coefficients.

Definition 4. *The Chebyshev polynomial* of degree j is denoted $T_j(t)$ and is given by the explicit formula

$$T_j(t) = \cos(j \arccos t).$$

Definition 5. *The shifted Chebyshev polynomial* is

$$T_j^*(t) = T_j(2t - 1).$$

At first glance this may look trigonometric, however Definition 5 can be combined

with trigonometric identities to yield:

$$\begin{aligned} T_0(t) &= 1, \\ T_1(t) &= 2t - 1, \\ T_2(t) &= 8t^2 - 8t + 1, \\ &\cdots \\ T_{j+1}(t) &= (4t-2)T_j(t) - T_{j-1}(t), \quad j \geq 1. \end{aligned}$$

The following facts about Chebyshev polynomials can be found in [NR] or [R]. We list them here with the only difference that we have translated those properties to shifted Chebyshev polynomials case.

From the definitions it follows that the domains of $T_j(t)$ and $T_j^*(t)$ are $[-1,1]$ and $[0,1]$ respectively, and they both range between -1 and 1. The shifted Chebyshev polynomials are orthogonal over a weight $(t-t^2)^{-1/2}$. In particular,

$$\int_0^1 \frac{T_i^*(t)T_j^*(t)}{\sqrt{t-t^2}}\,dt = \begin{cases} 0, & i \neq j \\ \pi, & i = j = 0, \\ \pi/2, & i = j = 1, 2, \ldots \end{cases} \tag{2.5}$$

The polynomial $T_j^*(t)$ has j zeros on $[0,1]$, and they are located at the points

$$t_k = 0.5 \cos \frac{\pi(k-\frac{1}{2})}{j} + 0.5 \quad k = 1, ..., j \tag{2.6}$$

The Chebyshev polynomials satisfy a discrete orthogonality relation:

$$\sum_{k=1}^{m} T_i^*(t_k)T_j^*(t_k) = \begin{cases} 0, & i \neq j \\ m, & i = j = 0, \\ m/2, & i = j = 1, 2, \ldots, m-1 \end{cases} \tag{2.7}$$

where t_k are m zeros of $T_m^*(t)$ given by (2.6).

Now that we have stated the main properties of Chebyshev polynomials it is time to discuss the way we use them to approximate a continuous function $f(t)$. That is we want to find coefficients c_js such that $f(t) = \sum_{j=0}^{\infty} c_j T_j^*(t)$. Moreover, we want to find a number of Chebyshev polynomials m such that $\sum_{j=0}^{m-1} c_j T_j^*(t)$ approximates $f(t)$ with a desired tolerance.

There are two ways to find coefficients c_js: the integral method (that is utilize (2.5)) and interpolation. Interpolation is a genuinely faster and more general method. Its idea is to find a polynomial of degree m that agrees with a function at m points, called *interpolation nodes.*

Given a number of coefficients m we can obtain a formula for the coefficients by combining Definition 5 and formulae (2.6) and (2.7). That is, if $f(t_k) = \sum_{j=0}^{m-1} c_j T_j^*(t_k)$ then

$$\sum_{k=1}^{m} T_j^*(t_k) f(t_k) = \sum_{j=0}^{m-1} c_j \sum_{k=1}^{m} T_j^*(t_k) T_i^*(t_k) = \begin{cases} \frac{m}{2} c_i & \text{if } j \neq 0 \\ m c_0 & \text{if } j = 0 \end{cases}$$

Therefore

$$c_j = \begin{cases} \frac{2}{m} \sum_{k=1}^{m} f(t_k) T_j^*(t_k) & \text{if } j \neq 0 \\ \frac{1}{m} \sum_{k=1}^{m} f(t_k) T_j^*(t_k) & \text{if } j = 0 \end{cases} \tag{2.8}$$

Definition 6.

$$P_f^{m-1}(t) = \sum_{j=0}^{m-1} c_j T_j^*(t),$$

where c_js are described by (2.8), is *an approximation polynomial* of degree $m-1$ which interpolation nodes are at zeros of $T_m^*(t)$.

What about the error of the Chebyshev approximation?

First, we need the fact that in the expansion of a nice function Chebyshev coefficients $\widetilde{c}_j$s decay rapidly and $\widetilde{c}_j \simeq c_j$, for $j = 0, \ldots, m-1$. By $\widetilde{c}_j$s we mean coefficients

obtained by the integral method. Then

$$|f(t) - P_f^{m-1}(t)| \simeq |\sum_{j=m}^{\infty} \widetilde{c}_j T_j^*(t)| \leq \sum_{j=m}^{\infty} |\widetilde{c}_j|, \tag{2.9}$$

since $|T_j^*(t)| \leq 1$, for any j and $t \in [0,1]$. Therefore the error of the approximation polynomial $P_f^{m-1}(t)$ is dominated by the term $\widetilde{c}_m$ and is an oscillatory function with m *equal* extrema distributed *uniformly* over the interval $[0,1]$ (see [R] for details). This smooth spreading out of the error is the property that we are after, since we would like to find an approximating polynomial which has very small deviation from the true function $f(t)$ *at each time.*

So how big is this deviation? Adjusting Corollary 8.11 of [BF] to shifted Chebyshev polynomials case we get the following:

Proposition 2. *If f has m continuous derivatives on $t \in [0,1]$ then*

$$\max_{t\in[0,1]} |f(t) - P_f^{m-1}(t)| \leq \frac{1}{2^{2m-1}m!} \max_{t\in[0,1]} |f^{(m)}(t)|.$$

In fact, all polynomial interpolation schemes have a factor of $\frac{1}{2^m m!}$ but the benefit of using shifted Chebyshev polynomials is to have a factor $\frac{1}{2^{2m-1}}$ which is as small as it can be for polynomial interpolation.

We will use the above proposition to estimate the number of shifted Chebyshev polynomials needed to approximate entries of $A(t)$ and $B(t)$. However, the question of how to find this number for accurate approximation of entries of $\Phi(t)$ and $\Psi(t)$ for noncommutative $A(t)$ is still open. If $A(t)$ is a commutative matrix we can use its spectral radius ρ to obtain an upper bound for an exponential growth of the entries of $\Phi(t)$ and $\Psi(t)$, that is set $f(t) = e^{\rho t}$ in Proposition 2. The proof of this fact follows directly from using Magnus expansion (see [M] and [I]). For further discussion see examples in Chapter 4.

2.4 Chebyshev approximation of fundamental solutions

Once we have figured out how to approximate a scalar function of time we can proceed to approximating of matrix-valued functions $\Phi(t)$ and $\Psi(t)$.

First, let us define some operations on matrices and their properties.

Definition 7. If $\mathbf{b}_{ij}$ are $m \times 1$ column vectors and $\mathbf{B}$ is the $nm \times n$ matrix

$$\mathbf{B} = \begin{pmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \dots & \mathbf{b}_{1n} \\ \mathbf{b}_{21} & \mathbf{b}_{22} & \dots & \mathbf{b}_{2n} \\ \dots & \dots & \dots & \dots \\ \mathbf{b}_{n1} & \mathbf{b}_{n2} & \dots & \mathbf{b}_{nn} \end{pmatrix}$$

then $\mathbf{B}'$ is the $n \times nm$ matrix

$$\mathbf{B}' = \begin{pmatrix} \mathbf{b}_{11}^\top & \mathbf{b}_{12}^\top & \dots & \mathbf{b}_{1n}^\top \\ \mathbf{b}_{21}^\top & \mathbf{b}_{22}^\top & \dots & \mathbf{b}_{2n}^\top \\ \dots & \dots & \dots & \dots \\ \mathbf{b}_{n1}^\top & \mathbf{b}_{n2}^\top & \dots & \mathbf{b}_{nn}^\top \end{pmatrix}.$$

Definition 8. Given a $q \times r$ matrix $\mathbf{A}$ and an $m \times n$ matrix $\mathbf{B}$ their Kronecker product is a $(qm) \times (rn)$ matrix

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} \mathbf{a}_{11}\mathbf{B} & \mathbf{a}_{12}\mathbf{B} & \dots & \mathbf{a}_{1r}\mathbf{B} \\ \mathbf{a}_{21}\mathbf{B} & \mathbf{a}_{22}\mathbf{B} & \dots & \mathbf{a}_{2r}\mathbf{B} \\ \dots & \dots & \dots & \dots \\ \mathbf{a}_{q1}\mathbf{B} & \mathbf{a}_{q2}\mathbf{B} & \dots & \mathbf{a}_{qr}\mathbf{B} \end{pmatrix}.$$

Definition 9. Given an $q \times r$ matrix $\mathbf{T}(t)$ define $\hat{\mathbf{T}}(t) = I \otimes \mathbf{T}(t)$.

Definition 10. If $\mathbf{M}$ is an $nm \times n$ matrix then $\mathbf{M}^\circ$ is an $nm \times n$ matrix defined as $\mathbf{M}^\circ = (\mathbf{M}')^\top = (\mathbf{M}^\top)'$.

Proposition 3. *If* $\mathbf{M} = A \otimes B$ *then* $\mathbf{M}' = A \otimes B^T$.

Proof. The result follows directly from the above definitions. □

Now let m be the number of shifted Chebyshev polynomials we use. Let $\mathbf{T}(t) = (T_0^*(t)\ T_1^*(t)\ \ldots\ T_{m-1}^*(t))^\top$ be an $m \times 1$ vector of the polynomials. Then we can rewrite the approximation polynomial for a scalar function $f(t)$ as

$$f(t) = \sum_{j=0}^{m-1} c_j T_j^*(t) = \mathbf{T}(t)^\top \mathbf{c}, \tag{2.10}$$

where $\mathbf{c}$ is the $m \times 1$ vector of coefficients.

Similar to (2.10) we can assign an $m \times 1$ vector of coefficients $\mathbf{a}_{ij}$ to each entry $a_{ij}(t)$ of $A(t)$. Then

$$\begin{aligned} A(t) &= \begin{pmatrix} \mathbf{T}(t)^\top \mathbf{a}_{11} & \mathbf{T}(t)^\top \mathbf{a}_{12} & \ldots & \mathbf{T}(t)^\top \mathbf{a}_{1n} \\ \mathbf{T}(t)^\top \mathbf{a}_{21} & \mathbf{T}(t)^\top \mathbf{a}_{22} & \ldots & \mathbf{T}(t)^\top \mathbf{a}_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{T}(t)^\top \mathbf{a}_{n1} & \mathbf{T}(t)^\top \mathbf{a}_{n2} & \ldots & \mathbf{T}(t)^\top \mathbf{a}_{nn} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{T}(t)^\top & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{T}(t)^\top & \ldots & \mathbf{0} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{T}(t)^\top \end{pmatrix} \cdot \begin{pmatrix} \mathbf{a}_{11} & \mathbf{a}_{12} & \ldots & \mathbf{a}_{1n} \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \ldots & \mathbf{a}_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{a}_{n1} & \mathbf{a}_{n2} & \ldots & \mathbf{a}_{nn} \end{pmatrix} = \hat{\mathbf{T}}(t)'\mathbf{A}, \end{aligned} \tag{2.11}$$

where $\mathbf{A}$ is an $nm \times n$ matrix of coefficients.

Similarly, we can expand $B(t)$ as

$$B(t) = \hat{\mathbf{T}}(t)'\mathbf{B}. \tag{2.12}$$

Note that if $\mathbf{M}$ is a matrix of coefficients for $M(t)$ then $\mathbf{M}^\circ$ is a matrix of coefficients for $M^\top(t)$.

Now we want to combine the Picard iteration formula for $\Phi^{(p)}(t)$ (2.2) and the expansion formula analogous to (2.11) and (2.12). For that we need tools for integrating and multiplying expanded matrix-valued functions.

Lemma 3. *If $M(t)$ is a continuous $n \times n$ matrix-valued function of $t \in [0,1]$, and if $M(t) = \hat{\mathbf{T}}(t)'\mathbf{M}$ then*

$$\int_0^t M(s)ds = \int_0^t \hat{\mathbf{T}}(s)'\mathbf{M}ds = \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{M} + O(\frac{1}{m}),$$

where $\mathbf{G} = I_n \otimes G$ is an $nm \times nm$ matrix and G is a constant matrix called integration operational matrix, described below. The error term $O(\frac{1}{m})$ is a matrix with entries bounded by $\frac{1}{4m} \max_{1 \le i,j \le m} |(\mathbf{m}_{ij})_m|$.

Proof. First we need to find an expansion formula for

$$\int_0^t \mathbf{T}^\top(s)ds = \int_0^t \begin{pmatrix} T_0^*(s) & T_1^*(s) & \dots & T_{m-1}^*(s) \end{pmatrix} ds.$$

Let's integrate the first two shifted Chebyshev polynomials:

$$\begin{aligned} \int_0^t T_0^*(s)ds &= t = \frac{1}{2}(T_1^*(t) + T_0^*(t)) \\ \int_0^t T_1^*(s)ds &= t^2 - t = \frac{1}{8}(T_2^*(t) - T_0^*(t)). \end{aligned} \tag{2.13}$$

We claim that

$$\int_0^t T_k^*(s)ds = \frac{T_{k+1}^*(t)}{4(k+1)} - \frac{T_{k-1}^*(t)}{4(k-1)} - \frac{(-1)^k T_0^*(t)}{2(k^2-1)}, \text{ for any } k \ge 2. \tag{2.14}$$

Let's check the base case first:

$$\int_0^t T_2^*(s)ds = \frac{T_3^*(t)}{12} - \frac{T_1^*(t)}{4} - \frac{T_0^*(t)}{6} = \frac{8}{3}t^3 - 4t^2 + t.$$

Now assume that our claim is true for $2 \le i \le k$ then using the recursive definition of Chebyshev polynomials and the fact that

$$(4t-2)T_k^*(t) = \begin{cases} T_{k+1}^*(t) + T_{k-1}^*(t) & \text{if } k \ge 2, \\ 2(T_2^*(t) + T_0^*(t)) & \text{if } k = 1, \\ 2T_1^*(t) & \text{if } k = 0 \end{cases}$$

by integration by parts we get

$$\int_0^t T_{k+1}^*(s)ds = \int_0^t [(4s-2)T_k^*(s) - T_{k-1}^*(s)]ds$$
$$= (4t-2)\int_0^t T_k^*(s)ds - 4\int_0^t \left[\frac{T_{k+1}^*(s)}{4(k+1)} - \frac{T_{k-1}^*(s)}{4(k-1)} - \frac{(-1)^k T_0^*(s)}{2(k^2-1)}\right] ds$$
$$- \frac{T_k^*(t)}{4k} + \frac{T_{k-2}^*(t)}{4(k-2)} + \frac{(-1)^{k-1}T_0^*(t)}{2(k^2-2k)}$$
$$= \frac{1}{4(k+1)}(T_{k+2}^*(t) + T_k^*(t)) - \frac{1}{4k}(T_k^*(t) + T_{k-2}^*(t)) - \frac{(-1)^k T_1^*(t)}{k^2-1}$$
$$- \frac{1}{k+1}\int_0^t T_{k+1}^*(s)ds + \frac{1}{k+1}\left[\frac{T_k^*(t)}{4k} - \frac{T_{k-2}^*(t)}{4(k-2)} - \frac{(-1)^{k-1}T_0^*(t)}{2(k^2-2k)}\right]$$
$$+ \frac{(-1)^k}{k^2-1}(T_1^*(t) + T_0^*(t)) - \frac{T_k^*(t)}{4k} + \frac{T_{k-2}^*(t)}{4(k-2)} + \frac{(-1)^{k-1}T_0^*(t)}{2(k^2-2k)}.$$

Combining the unknown integrals on the left side of the equation we get:

$$\frac{k+2}{k+1}\int_0^t T_{k+1}^*(s)ds$$
$$= \frac{1}{4(k+1)}T_{k+2}^*(t) + \left[\frac{1}{4(k+1)} - \frac{1}{4(k-1)} + \frac{1}{4k(k-1)} - \frac{1}{4k}\right] T_k^*(t)$$
$$+ (-1)^k\left[\frac{1}{2k(k-1)(k-2)} + \frac{1}{k^2-1} - \frac{1}{2k(k-2)}\right] T_0^*(t).$$

So

$$\int_0^t T_{k+1}^*(s)ds = \frac{T_{k+2}^*(t)}{4(k+2)} + \frac{T_k^*(t)}{4k} - \frac{(-1)^{k+1}T_0^*(t)}{2((k+1)^2-1)}.$$

Therefore, our claim is true.

Using (2.13) and (2.14) we can now obtain the expansion formula for $\int_0^t \mathbf{T}^\top(s)ds$:

$$\int_0^t \mathbf{T}^\top(s)ds \simeq \mathbf{T}^\top(t)\begin{pmatrix} \frac{1}{2} & -\frac{1}{8} & -\frac{1}{6} & \cdots & \cdots & \frac{(-1)^m}{2m(m-2)} \\ \frac{1}{2} & 0 & -\frac{1}{4} & \cdots & \cdots & 0 \\ 0 & \frac{1}{8} & 0 & \cdots & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & \cdots & -\frac{1}{4(m-2)} \\ 0 & 0 & 0 & \cdots & \frac{1}{4m} & 0 \end{pmatrix} = \mathbf{T}^\top(t)G^\top.$$

In general, an entry on the i^{th} row and j^{th} column of G is calculated according to the following formula:

$$g_{ij} = \begin{cases} \frac{1}{2} & \text{if } i = 1,\ j = 1, 2 \\ -\frac{1}{8} & \text{if } i = 2,\ j = 1 \\ \frac{(-1)^i}{2i(i-2)} & \text{if } i = 2,\ i > 2 \\ \frac{1}{4i} & \text{if } j \geq 2,\ i \geq 2,\ i = j - 1 \\ -\frac{1}{4(i-2)} & \text{if } j \geq 2,\ i \geq 2,\ i = j + 1 \\ 0 & \text{otherwise} \end{cases}.$$

Note that we do not get equality in the expansion formula since we are chopping the $T_m^*(t)$ term, which appears in integration formula for $T_{m-1}^*(t)$, in order to get $m \times m$ matrix. Since $|T_m^*(t)| \leq 1$, for any $t \in [0, 1]$ the truncation error is bounded by $\frac{1}{4m}$.

So we get the following expansion formula for the integral of a matrix-valued function:

$$\int_0^t M(s)ds \simeq \begin{pmatrix} \mathbf{T}^\top(t)G^\top & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{T}^\top(t)G^\top & \dots & \mathbf{0} \\ \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{T}^\top(t)G^\top \end{pmatrix} \mathbf{M} = \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{M}.$$

□

Lemma 4. *If $R(t), S(t)$ are two continuous $n \times n$ matrix-valued functions of $t \in [0, 1]$, and if $R(t) = \hat{\mathbf{T}}(t)'\mathbf{R}$, $S(t) = \hat{\mathbf{T}}(t)'\mathbf{S}$, and if the (i, j) entry of $R(t)$ is expanded as $r_{ij}(t) = \mathbf{T}(t)^\top \mathbf{r}_{ij}$, then*

$$R(t)S(t) = \hat{\mathbf{T}}(t)'\mathbf{R}\hat{\mathbf{T}}(t)'\mathbf{S} = \mathbf{R}'\hat{\mathbf{T}}(t)\hat{\mathbf{T}}(t)'\mathbf{S} \simeq \hat{\mathbf{T}}(t)'\mathbf{Q}_R\mathbf{S}$$

where $\mathbf{Q}_R$ is an $nm \times nm$ matrix whose (i, j)th $m \times m$ submatrix is $\mathbf{Q}_{r_{ij}}$, which is the matrix displayed as (12) in [SB]. In particular, $R(t)I = \hat{\mathbf{T}}(t)'\mathbf{Q}_R\mathbf{I} = \hat{\mathbf{T}}(t)'\mathbf{R}$.

From now on we pretend the use of $\mathbf{G}$ and $\mathbf{Q}$ is exact. From Lemma 3 it follows that

$$(G_A I)(t) = \int_0^t A(s)\,ds = \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{A}.$$

Then using Lemma 4 we get

$$(G_A(G_A I))(t) = \int_0^t A(s_1) \int_0^{s_1} A(s_0)\,ds_0\,ds_1 = \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{Q}_A\mathbf{G}'\mathbf{A},$$

and so on.

If $W_A = \mathbf{Q}_A\mathbf{G}'$ then it can be shown by induction that

$$\underbrace{(G_A(G_A(\ldots(G_A}_{p \text{ times}} I)\ldots)))(t) = \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{W}_A^p\mathbf{A}.$$

Now we can write an expansion formula for $\Phi^{(p)}(t)$:

$$\Phi^{(p)}(t) = \hat{\mathbf{T}}(t)'\mathbf{\Phi} = \hat{\mathbf{T}}(t)'(\mathbf{I} + \mathbf{G}'\mathbf{A} + \mathbf{G}'\mathbf{W}_A\mathbf{A} + \cdots + \mathbf{G}'\mathbf{W}_A^p\mathbf{A}). \tag{2.15}$$

In more computationally efficient Horner's form $\mathbf{\Phi}$ looks like:

$$\mathbf{\Phi} = \mathbf{I} + \mathbf{G}'(\mathbf{A} + \mathbf{W}_A(\mathbf{A} + \cdots + \mathbf{W}_A(\mathbf{A} + \mathbf{W}_A\mathbf{A})\ldots)).$$

Similarly, we can obtain an expansion formula for $\Psi^{(p)}$:

$$\begin{aligned}\Psi^{(p)}(t) &= \hat{\mathbf{T}}(t)'\mathbf{\Psi} \\ &= \hat{\mathbf{T}}(t)'(\mathbf{I} - \mathbf{G}'(\mathbf{A}^\circ - \mathbf{W}_{A^\top}(\mathbf{A}^\circ - \cdots - \mathbf{W}_{A^\top}(\mathbf{A}^\circ - \mathbf{W}_{A^\top}\mathbf{A}^\circ)\ldots))),\end{aligned} \tag{2.16}$$

where $\mathbf{W}_{A^\top} = \mathbf{Q}_{A^\top}$.

2.5 Approximation of delay Floquet transition matrix

Now we have all the tools and results necessary to write down the expansion formula for $(Ux)(t) \simeq \hat{\mathbf{T}}(t)'\mathbf{u}$. We will just need one more expansion for the initial condition

vector $x(t) = \hat{\mathbf{T}}(t)'\mathbf{v}$. So (1.9) becomes:

$$\begin{aligned}
\hat{\mathbf{T}}(t)'\mathbf{u} &= \hat{\mathbf{T}}(t)'\mathbf{\Phi}\left\{\hat{\mathbf{T}}(1)'\mathbf{v} + \int_0^t (\hat{\mathbf{T}}(s)'\mathbf{\Psi})^\top \hat{\mathbf{T}}(s)'\mathbf{B}\hat{\mathbf{T}}(s)'\mathbf{v}\, ds\right\} \\
&= \hat{\mathbf{T}}(t)'\mathbf{\Phi}\left\{\hat{\mathbf{T}}(1)'\mathbf{v} + \int_0^t (\mathbf{\Psi}^\circ)'\hat{\mathbf{T}}(s)\hat{\mathbf{T}}(s)'\mathbf{B}\hat{\mathbf{T}}(s)'\mathbf{v}\, ds\right\} \\
&\overset{\text{Lemma 4}}{=} \hat{\mathbf{T}}(t)'\mathbf{\Phi}\left\{\hat{\mathbf{T}}(1)'\mathbf{v} + \int_0^t \hat{\mathbf{T}}(s)'\mathbf{Q}_{\Psi^\top}\mathbf{B}\hat{\mathbf{T}}(s)'\mathbf{v}\, ds\right\}.
\end{aligned}$$

Let $\mathbf{Z} = \mathbf{Q}_{\Psi^\top}\mathbf{B}$ and let $Z(t) = \hat{\mathbf{T}}(t)'\mathbf{Z}$ then

$$\begin{aligned}
\hat{\mathbf{T}}(t)'\mathbf{u} &= \hat{\mathbf{T}}(t)'\mathbf{\Phi}\left\{\hat{\mathbf{T}}(1)'\mathbf{v} + \int_0^t (\hat{\mathbf{T}}(s)\mathbf{Q}_Z\mathbf{v}\, ds\right\} \\
&\overset{\text{Lemma 3}}{=} \hat{\mathbf{T}}(t)'\mathbf{\Phi}\left\{\hat{\mathbf{T}}(1)'\mathbf{v} + \hat{\mathbf{T}}(t)'\mathbf{G}'\mathbf{Q}_Z\mathbf{v}\right\} \\
&\overset{\text{Lemma 4}}{=} \hat{\mathbf{T}}(t)'\left\{\mathbf{\Phi}\hat{\mathbf{T}}(1)' + \mathbf{Q}_\Phi\mathbf{G}'\mathbf{Q}_Z\right\}\mathbf{v}.
\end{aligned}$$

Therefore,

$$\mathbf{W} = \mathbf{\Phi}\hat{\mathbf{T}}(1)' + \mathbf{Q}_\Phi\mathbf{G}'\mathbf{Q}_Z \tag{2.17}$$

is a $nm \times nm$ matrix of shifted Chebyshev coefficients for $(Ux)(t)$.

Note that dimension of $\mathbf{W}$ need not be $nm \times nm$ in practice. Since U acts on space $V = C[0,1]$ we need to approximate this space as well. However, choosing the number of shifted Chebyshev polynomials l to approximate V is a separate issue from choosing the number of polynomials m to approximate the fundamental solution. If we assume that $l \leq m$ then after we obtain $nm \times nm$ matrix $\mathbf{W}$ we would want to truncate it to $nl \times nl$ matrix. This truncation is very helpful in further analysis of stability since by reducing the size of $\mathbf{W}$ appropriately we decrease computation time with very little loss of accuracy.

A couple of words need to be said about how we truncate $nm \times nm$ matrix $\mathbf{W}$. We can view it as a matrix consisting of n^2 blocks of size $m \times m$. Each block contributes to a different part of the solution and therefore we need to truncate each block. That is, if we want to obtain an $nl \times nl$ approximation to $\mathbf{u}$ we need to cut $l \times l$ blocks from the upper left corners of which are elements of the following set: $\{\mathbf{u}_{rq} \mid r, q = 1, m+1, 2m+1, ..., (n-1)m+1\}$.

2.6 Alternative idea

We have decided to approximate the solutions to an ODE system

$$Y'(t) = A(t)Y(t), \quad Y(0) = I \tag{2.18}$$

by Picard iterations. However, this is not the only and probably not the fastest approach to the problem. Another idea is to represent the solution of (2.18) in the form $Y(t) = e^{\Omega(t)}$ and then rephrase the differential equation in terms of $\Omega(t)$. Wilhelm Magnus (see [M]) showed that

$$\begin{aligned}\Omega(t) = &\int_0^t A(s)ds - \frac{1}{2}\int_0^t \int_0^{s_1} [A(s_2), A(s_1)]ds_2 ds_1 \\ &+ \frac{1}{12}\int_0^t \int_0^{s_1} \int_0^{s_2} [A(s_3), [A(s_2), A(s_1)]]ds_3 ds_2 ds_1 + \dots\end{aligned} \tag{2.19}$$

There are two main advantages of this method. First, it will require less iterations due to a much better initial "guess" of the solution compared to Picard iterations. That is if $A(t)$ is a commutative matrix (and therefore $[A(s_1), A(s_2)] = 0$, for any s_1, s_2) other than identity we will still need some number of Picard iterations to find $Y(t)$ but in Magnus series (2.19) the very first term is *exactly* $\Omega(t)$.

Another advantage of Magnus expansion is that in numerical computation it showed significantly smaller error compared to other numerical schemes such as Runge-Kutta and Gauss-Legendre Runge-Kutta (see [I]).

Chapter 3

Analysis of stability

3.1 Introduction

In Chapter 1 we concluded that the stability question comes down to the question of whether eigenvalues of DFTM are inside of the unit circle. The problem at the moment is that, first, U is infinite-dimensional operator, and second, U does not have an analytical expression. In Chapter 2 we have derived the approximation matrix $\mathbf{W}$ of shifted Chebyshev coefficients for U. In [BBA] the reader can find the arguments about why the stability question is approximately answered in terms of the eigenvalues of $nl \times nl$ matrix $\mathbf{W}$. In this chapter we will discuss four stability criteria: Routh-Hurwitz, Schur-Cohn, Jury-Marden, and Schur-Cohn-Fujiwara. All of them address the stability nature of $\mathbf{W}$ in terms of the coefficients of its characteristic polynomial. The main reason for choosing these criteria was that none of them uses division which is a big obstacle in symbolic computations. The description of these (and more) methods was found in [B] and the reader is advised to look up proofs and examples in there.

All stability criteria are divided into two groups: continuous-time case (characteristic polynomial is stable if and only if its zeros are in the left half-plane) and

discrete-time case (characteristic polynomial is stable if and only if its zeros are inside of the unit circle). Note that a square matrix A has eigenvalues in the left-half plane if and only if e^A has eigenvalues in the unit circle. First group is represented by Routh-Hurwitz method and second one is represented by the other three methods.

Since the characteristic polynomial of $\mathbf{W}$ falls into the discrete-time category, the reader might wonder why we even bother to consider Routh-Hurwitz method. However, on practice it turned out to be the fastest method among the presented four. One can transform between the categories by the substitution $\lambda = \frac{\mu+1}{\mu-1}$.

Together with the stability criteria we will also give the root location criteria (relative to the imaginary axis or the unit circle) for the sake of integrity of information. Writing an appropriate piece of code for the root location is one of the future goals of the project.

3.2 Obtaining the characteristic polynomial

Definition 11. The characteristic polynomial of an $nl \times nl$ matrix $\mathbf{A}$ is

$$p(\lambda) = \det(\lambda I_n - \mathbf{A}) = \lambda^n + \alpha_1 \lambda^{n-1} + \cdots + \alpha_{n-1}\lambda + \alpha_n.$$

Since all we need is the *coefficients* of the characteristic polynomial of $\mathbf{W}$, using the definition does not seem to be the best way to get it. Matrix $\mathbf{W}$ already contains one or more parameters. Introducing a new parameter λ will obviously complicate symbolic operations such as **Expand**[1]. Moreover λ is a temporary parameter and dropping it from the polynomial is time-consuming as well.

There is another more straightforward method to find coefficients of characteristic polynomial using traces (see [CRC]):

[1] In practice, *Mathematica* deals better with expressions in expanded form, unless there is a specific form known, like Horner's form of a polynomial, for plotting. This is why we keep expanding while computing matrices such as $\mathbf{\Phi}$, $\mathbf{\Psi}$, and $\mathbf{W}$.

Definition 12. The characteristic polynomial of an $nl \times nl$ matrix $\mathbf{A}$ can be obtained as:

$$\lambda^{nl} - (tr\mathbf{A})\lambda^{nl-1} + (tr_2\mathbf{A})\lambda^{nl-2} - \cdots + (-1)^{nl-1}(tr_{nl-1}\mathbf{A})\lambda + (-1)^{nl}\det\mathbf{A}, \quad (3.1)$$

where $tr_k\mathbf{A}$ is the sum of all the determinants of all $\binom{nl}{k}$ submatrices of $\mathbf{A}$ of size $k \times k$:

$$tr_k\mathbf{A} = \sum_{1 \le i_1 < i_2 < \cdots < i_k \le nl} \begin{vmatrix} a_{i_1,i_1} & a_{i_1,i_2} & \cdots & a_{i_1,i_k} \\ a_{i_2,i_1} & a_{i_2,i_2} & \cdots & a_{i_2,i_k} \\ \cdots & \cdots & \cdots & \cdots \\ a_{i_k,i_1} & a_{i_k,i_2} & \cdots & a_{i_k,i_k} \end{vmatrix} \quad (3.2)$$

In any case we obtain a parameter-dependent $1 \times (nl+1)$ vector of coefficients of the characteristic polynomial $p(\lambda)$ of $\mathbf{W}$:

$$\vec{\alpha} = (1, \alpha_1, \alpha_1, \ldots, \alpha_{nl}). \quad (3.3)$$

3.3 Routh-Hurwitz criterion

Routh-Hurwitz criteion is designed for a continuous-time characteristic polynomial. Thus in order to apply it to $\vec{\alpha}$ we need to perform a transformation $\lambda = \frac{\mu+1}{\mu-1}$ which maps the unit disk to the left half plane. Again for the reasons similar to those mentioned in 3.2 we would like a faster transformation technique than introducing a new variable (μ), simplifying the expression, and then dropping the variable.

In [B] it is shown that the transformation of the coefficients can be performed with the aid of constant coefficient $(nl+1) \times (nl+1)$ matrix $\Gamma = [\gamma_{ij}]$, $i, j = 1, \ldots, nl+1$ where

$$\gamma_{ij} = \begin{cases} 1 & \text{if } i = 1, \ldots, nl+1,\ j = nl+1 \\ (-1)^{nl-j+1}\binom{nl}{j-1} & \text{if } i = 1,\ j = 1, \ldots, nl+1, \\ \gamma_{i,j+1} + \gamma_{i-1,j+1} + \gamma_{i-1,j} & \text{otherwise} \end{cases} \quad (3.4)$$

So we get an $1 \times (nl+1)$ vector of coefficients of the discrete-time characteristic polynomial $\widetilde{p}(\mu)$ of $\mathbf{W}$:

$$\overrightarrow{\beta} = (\alpha_{nl}, \alpha_{nl-1}, \ldots, \alpha_1, 1)\Gamma_{nl+1} = (\beta_0, \beta_1, \beta_2, \ldots, \beta_{nl}). \tag{3.5}$$

Definition 13. An $n \times n$ matrix A is called *positive definite* if and only if the determinants of all upper-left submatrices are positive.

Theorem 2. *(Routh-Hurwitz)*

For (real) continuous-time characteristic polynomial $\widetilde{p}(\mu) = \beta_0\mu^{nl} + \beta_1\mu^{nl-1} + \cdots + \beta_{nl-1}\mu + \beta_{nl}$ *with* $\beta_0 > 0$ *construct the* $nl \times nl$ *Routh-Hurwitz matrix*

$$RH = \begin{pmatrix} \beta_1 & \beta_3 & \beta_5 & \ldots & \beta_{2nl-1} \\ \beta_0 & \beta_2 & \beta_4 & \ldots & \beta_{2nl-2} \\ 0 & \beta_1 & \beta_3 & \ldots & \beta_{2nl-3} \\ 0 & \beta_0 & \beta_2 & \ldots & \beta_{2nl-4} \\ 0 & 0 & \beta_1 & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & \beta_{nl} \end{pmatrix}$$

where $\beta_r = 0$, $r > nl$.

STABILITY CRITERIA *The polynomial* $\widetilde{p}(\mu)$ *is stable if and only if* RH *is positive definite.*

ROOT LOCATION *Provided that determinants of the upper-left submatrices of* RH, RH_i, *are nonzero for all* i, *then* $\widetilde{p}(\mu)$ *has* k *and* $nl - k$ *roots with positive and negative real parts, respectively,* k *being the number of sign variations in the sequences* $\{\beta_0, |RH_1|, |RH_3|, \ldots\}$ *and* $\{1, |RH_2|, |RH_4|, \ldots\}$.

3.4 Schur-Cohn, Jury-Marden, and Schur-Cohn-Fujiwara criteria

Theorem 3. *(Schur-Cohn)*

For (real) discrete-time characteristic polynomial $p(\lambda) = \alpha_0\lambda^{nl} + \alpha_1\lambda^{nl-1} + \cdots + \alpha_{nl-1}\lambda + \alpha_{nl}$ *with* $\alpha_0 > 0$ *construct the* $2nl \times 2nl$ *Schur-Cohn matrix*

$$SC = \begin{pmatrix} \Delta_1 & \Delta_2 \\ \Delta_2 & \Delta_1 \end{pmatrix}, \tag{3.6}$$

$$\Delta_1 = \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \dots & \alpha_{nl-1} \\ 0 & \alpha_0 & \alpha_1 & \dots & \alpha_{nl-2} \\ 0 & 0 & \alpha_0 & \dots & \alpha_{nl-3} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \alpha_0 \end{pmatrix} \tag{3.7}$$

$$\Delta_2 = \begin{pmatrix} 0 & \dots & 0 & 0 & \alpha_{nl} \\ 0 & \dots & 0 & \alpha_{nl} & \alpha_{nl-1} \\ 0 & \dots & \alpha_{nl} & \alpha_{nl-1} & \alpha_{nl-2} \\ \dots & \dots & \dots & \dots & \dots \\ \alpha_{nl} & \dots & \alpha_3 & \alpha_2 & \alpha_1 \end{pmatrix} \tag{3.8}$$

and denote by $\Delta^{(2nl-2k)}$ *the* $2(nl-k) \times 2(nl-k)$ *centrally situated submatrix of* SC.

STABILITY CRITERIA *The polynomial* $p(\lambda)$ *is stable if and only if* $|\Delta^{(2i)}| > 0$, $i = 1, 2, \dots, nl$.

ROOT LOCATION *Provided that* $|\Delta^{(2i)}| \neq 0$, *for all* i, *then* $p(\lambda)$ *has* k *and* $nl - k$ *roots inside and outside the unit disk, where* $nl - k$ *is the number of sign variations in the sequence* $\{1, |\Delta^{(2)}|, |\Delta^{(4)}|, \dots, |\Delta^{(2nl)}|\}$.

Theorem 4. *(Jury-Marden)*

For (real) discrete-time characteristic polynomial $p(\lambda) = \alpha_0\lambda^{nl} + \alpha_1\lambda^{nl-1} + \cdots + \alpha_{nl-1}\lambda + \alpha_{nl}$ *with* $\alpha_0 > 0$ *construct an array with an initial row*

$$\{c_{11}, c_{12}, \ldots, c_{1,nl+1}\} = \{\alpha_0, \alpha_1, \ldots, \alpha_{nl}\} \tag{3.9}$$

and subsequent rows defined by

$$c_{ij} = \det \begin{vmatrix} c_{i-1,1} & c_{i-1,j+1} \\ c_{i-1,nl-i+3} & c_{i-1,nl-j-i+2} \end{vmatrix}, \quad i = 2, 3, \ldots, nl+1 \tag{3.10}$$

STABILITY CRITERIA *The polynomial* $p(\lambda)$ *is stable if and only if* $c_{2,nl} > 0$, $c_{nl-i+3,1} < 0$, $i = 2, 3, \ldots, nl$.

ROOT LOCATION *Provided that* $c_{i,1} \neq 0$, *for all* i, *then* $p(\lambda)$ *has no roots with unit modulus, and there are* k *and* $nl - k$ *roots inside and outside the unit disk, where* k *is the number of negative products in the sequence* $\{(-1)^k c_{2,nl}c_{3,nl-1}\cdots c_{k+1,nl-k+1} |\ k = 1, 2, \ldots, nl\}$.

Theorem 5. *(Schur-Cohn-Fujiwara)*

STABILITY CRITERIA *The polynomial* $p(\lambda)$ *with positive leading coefficient is stable if and only if the symmetric Schur-Cohn matrix* $SCF = [k_{ij}]$, $i, j = 1, \ldots, nl$ *defined by*

$$k_{ij} = \sum_{r=0}^{i-1} (\alpha_{i-1-r}\alpha_{j-1-r} - \alpha_{nl+r-i+1}\alpha_{nl+r-j+1}), \quad i \leq j \tag{3.11}$$

is positive definite.

ROOT LOCATION *Provided that all determinants associated with upper-left submatrices of* SCF, K_i, *are nonzero for all* i, *then* $p(\lambda)$ *has* k *and* $nl - k$ *roots inside and outside the unit disk, where* $nl - k$ *is the number of sign variations in the sequence* $\{1, |K_1|, |K_2|, \ldots, |K_{nl}|\}$.

3.5 Comparison of the criteria

So which criterion is the best for our purposes? Let us first sum up all the features of the four criteria presented above that are important for symbolic computations in

Table 3.1: Stability criteria comparison.

Criterion	Coefficient transformation	Size of the matrix	Recursive procedure	Complicated entries
Routh-Hurwitz	Yes	$nl \times nl$	No	No
Schur-Cohn	No	$2nl \times 2nl$	No	No
Jury-Marden	No	2×2	Yes	No
Schur-Cohn-Fujiwara	No	$nl \times nl$	No	Yes

terms of computation time (see Table 3.1).

Note that entries in all matrices are complicated in the sense that they are polynomials of one or more parameters. By complicated entries here we mean those that contain products of the coefficients of the characteristic polynomial.

It is very difficult to sort the columns of Table 3.1 by their time share in the whole process of stability analysis. The most important one is definitely the size of a matrix. Since we are finding determinants in each criterion, the determinant of a $2nl \times 2nl$ matrix will involve 4 times more summations than the one of an $nl \times nl$ matrix, and each term of that sum will have twice as many factors. It is a huge complication for symbolic computations. Therefore, even though Schur-Cohn criteria is "the nicest" method in terms of the other three categories, it is the worst choice on the whole.

Jury-Marden matrices are only 2×2, but due to the recursive nature of the method their entries become more and more complicated with each iteration. The accumulation of the chopping error[2] also presents a complication.

Complicated entries in Schur-Cohn-Fujiwara matrix sufficiently slow down the

[2] **Chop**[*expr*, *error*] will drop all terms from the *expr* which numerical coefficients are smaller than the *error*. Chopping as well as expanding speeds up *Mathematica* calculations. However, chopping has to be done very carefully. For example, we can neglect term $10^{-20}a^{14}b^4$ with 6-digit accuracy if $0 \leq a \leq 10$, $0 \leq b \leq 1$. However if, say, $0 \leq a \leq 20$, $0 \leq b \leq 1$ this term can be as large as $2^{14}10^{-6}$ and therefore should not be dropped.

computation of the method.

In practice Routh-Hurwitz method proved to be the fastest among the presented four methods. It does require additional computation (transformation of parameters) but it is relatively fast. However, even this method fails to yield the results in reasonable amount of time for $nl \gtrsim 14$ which forced us to present numerical stability charts for a couple examples in Chapter 4 instead of the symbolic ones. What is more important, this prevents us from getting the symbolic boundaries for the problems we are interested in the project, such as turning problem.

3.6 Alternative ideas

While computing $\mathbf{W}$ is a matter of minutes even for large m and p, time for analysis of its stability grows with m and is a matter of weeks for $nl \gtrsim 14$. The main reason is that each criterion requires *a lot of* expanding and inevitably has to deal with very large degree polynomials. For example, if $A(t) = a^2$ and $m = 5$, $p = 10$ then the polynomials in $\mathbf{W}$ has a degree of a as high as 20.

So the major concern in the project at the moment is how to overcome this problem. We do not have a positive solution yet but we offer several ideas here that seem promising.

IDEA 1. Numerical computations with symbolic answer.

Since entries of $\mathbf{W}$ are polynomials of one or more parameters, the analysis of stability of $\mathbf{W}$ is based on polynomial multiplication and addition. However, we do not need to perform these operations symbolically in order to get a symbolic answer. We can use lists of coefficients instead. Then multiplication/addition of polynomials becomes convolution/addition of their coefficient lists. So as a result we will get lists of coefficients which can be easily translated to the appropriate polynomials of parameters. This idea requires *only* numerical computations and can be more effectively implemented in *Fortran* or *Matlab*. Note that it also requires a

new algorithm for computing determinants.

IDEA 2. Approximate stability results with Chebyshev polynomials.

Suppose we have two parameters, a and b, and we choose Schur-Cohn-Fujiwara criterion for stability analysis. Then we can use a finite set of two-dimensional shifted Chebyshev polynomials $T_j^*(a)T_k^*(b)$, $0 \leq j \leq m_a - 1$, $0 \leq k \leq m_b - 1$ to approximate the symbolic boundaries. For that first find zeros of the polynomials, (x_j, y_k). Then for each zero evaluate nl matrices K_is described in (3.11). Finally, for each i fit in the two-dimensional shifted Chebyshev polynomial with interpolation nodes at (x_j, y_k).

Chapter 4

Results

4.1 Introduction

Now that we have created a theory on how to approximately find the DFTM U we would want to see how well this theory works in practice. In the following sections we compare symbolic and numerical stability charts obtained for scalar constant DDE discussed in 1.2 and for Mathieu equations with exact results or results of other computations. We also demonstrate the convergence of the approximate solution to the exact one as m and p increase and the effect of truncating $\mathbf{W}$ on its eigenvalues and the solution of a given system. We also suppose the desired accuracy for approximation of the number of Picard iterations and the number of Chebyshev polynomials is 10^{-6}. That is we want to approximate entries of fundamental solution matrix with this accuracy.

4.2 Method of steps

Consider the following π-periodic system:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix}(t) = \begin{pmatrix} -\sin^2 t & 1 - 0.5\sin 2t \\ -1 - 0.5\sin 2t & -\cos^2 t \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}(t) + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}(t-\pi), \quad (4.1)$$

$$\begin{pmatrix} x \\ y \end{pmatrix}(t) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

The matrix of coefficients $A(t)$ in front of the non-delay term is the matrix

$$\begin{pmatrix} -1 + \alpha\cos^2 t & 1 - \alpha\sin t\cos t \\ -1 - \alpha\sin t\cos t & -1 + \alpha\sin^2 t \end{pmatrix}, \text{ where } \alpha = 1.$$

Even though $A(t)$ is not commutative the exact fundamental solution matrix $\Phi(t)$ is known [SB]. Therefore we can find the exact solution by method of steps and (1.6). We want to compare the exact solution of (4.1) obtained by using method of steps and the solution we get by iterative application of the matrix $\mathbf{W}$.

Before calculating $\mathbf{W}$ we need to decide on the number of Picard iterations p, the number of Chebyshev polynomials m needed to achieve the desired accuracy, and size l of $\mathbf{W}$.

In order to use Lemma 2 let's find $\overline{a}$ (using Asymptotically True Lemma):

$$\overline{a} = \max_{0\le t\le\pi} \pi\rho_A(t) = \pi\left|\frac{-1 \pm i\sqrt{3}}{2}\right| = \pi.$$

So we have $\sum_{q=p+1}^{\infty} \frac{\pi^q}{q!} \le 10^{-6} \Rightarrow p \ge 16$. Therefore let $p = 16$.

An upper bound for the m^{th} derivative of any entry of $A(t)$ is: $\pi \cdot (2\pi)^{m-1} = 2^{m-1}\pi^m$.

So according to Proposition 2 we have $\frac{\pi^m}{2^m m!} \le 10^{-6} \Rightarrow m \ge 12$. Therefore let $m = 12$. Let $l = 12$ (that is $\mathbf{W}$ is 24 x 24 matrix). Now we can calculate $\mathbf{W}$ and obtain the approximate solution to (4.1) by method of steps described in 1.4 (see Figure 4.1). The L_∞ error on the interval $[0, 9]$ is $\approx \binom{1.26\cdot 10^{-5}}{9.33\cdot 10^{-6}}$.

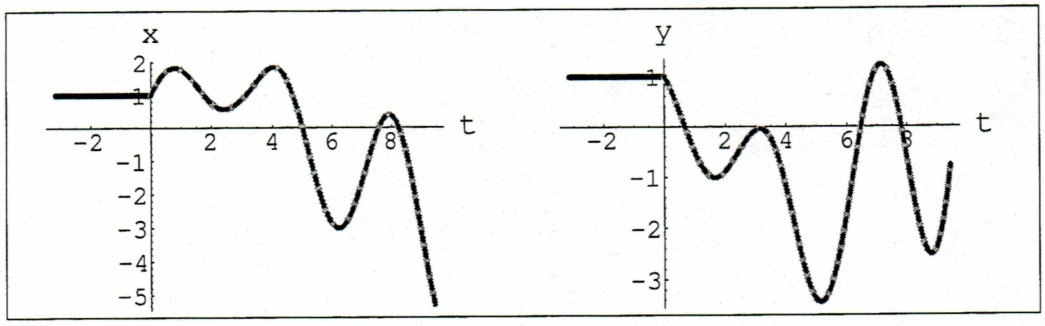

Figure 4.1: Initial condition (solid black), exact (solid grey) and approximate (dashed) solutions to (4.1) for $m = 12$, $p = 16$, $l = 12$.

Figures 4.2, 4.3, and 4.4 demonstrate the convergence of the approximate solution as the number m of Chebyshev polynomials, the number p of Picard iterations, and size l of $\mathbf{W}$ increases.

4.3 Change of eigenvalues

Since our major concern is stability of U, we would like to study behavior and convergence of the eigenvalues of its approximation matrix $\mathbf{W}$. Let's consider equation (4.1) again. As in the previous section let $m = 12$ and $p = 16$. Figure 4.5 demonstrates the behavior of the eigenvalues of $\mathbf{W}$.

On the Figure 4.6 we can see how spectral radius (largest in magnitude eigenvalue of W) change. Note that up to $l = 6$ it stays approximately the same, which means that the size of $\mathbf{W}$ could be reduced from 24×24 to only 12×12! Note, that Figure 4.4 corroborates this fact.

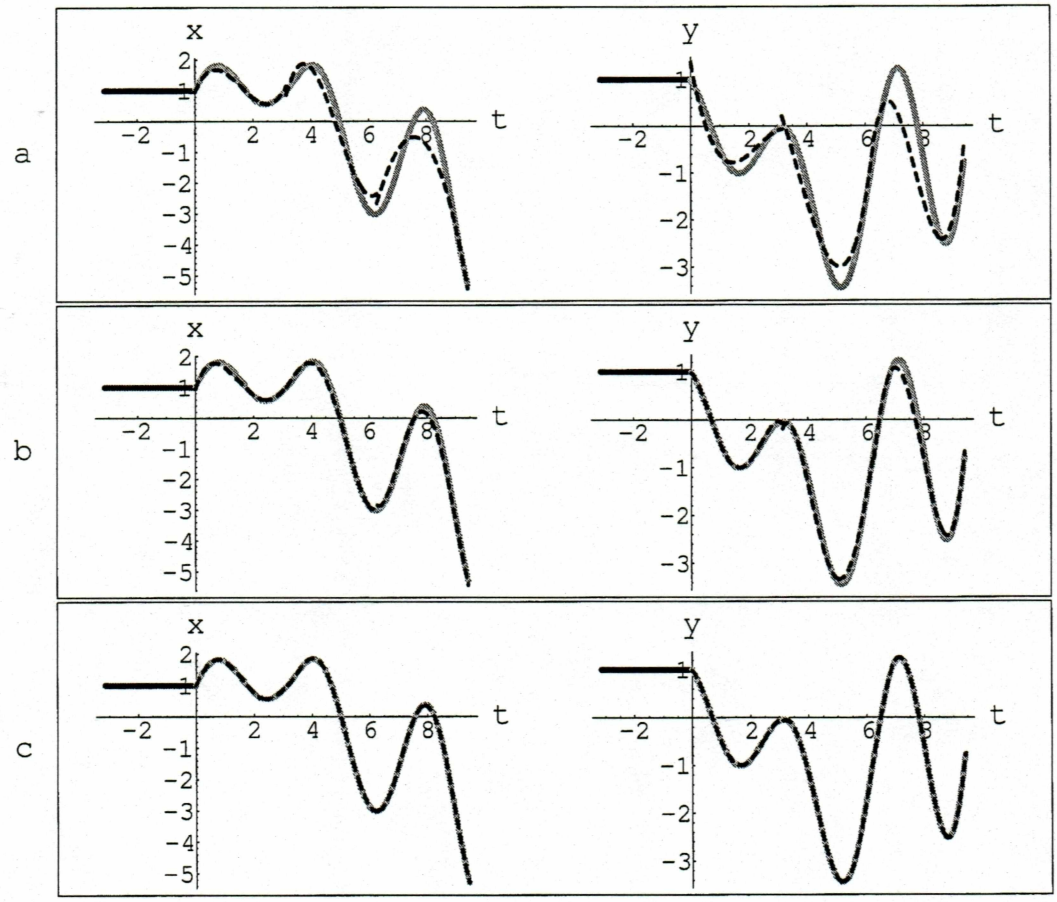

Figure 4.2: Initial condition (solid black), exact (solid grey) and approximate (dashed) solutions to (4.1) for $p = 16$, $l = m$, and $m =$ (a) 4; (b) 5; (c) 6.

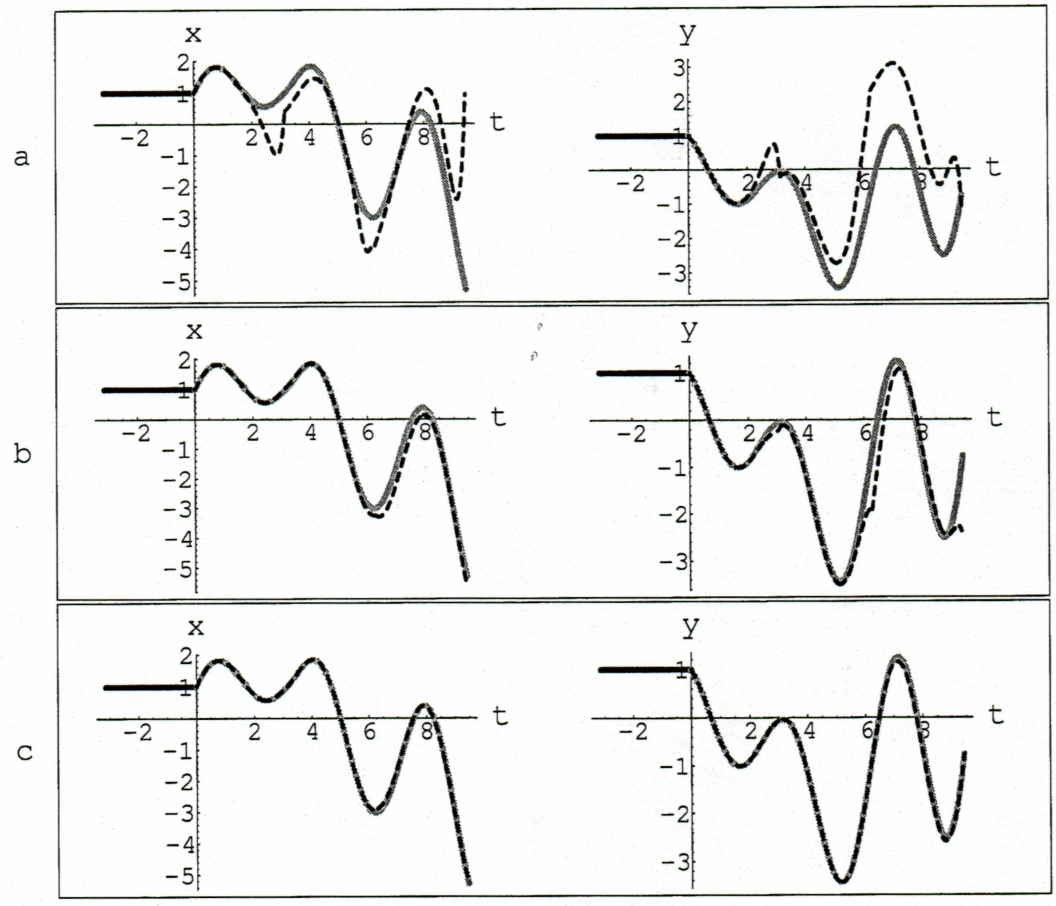

Figure 4.3: Initial condition (solid black), exact (solid grey) and approximate (dashed) solutions to (4.1) for $m = 12$, $l = 12$, and $p = (a)$ 6; (b) 8; (c) 10.

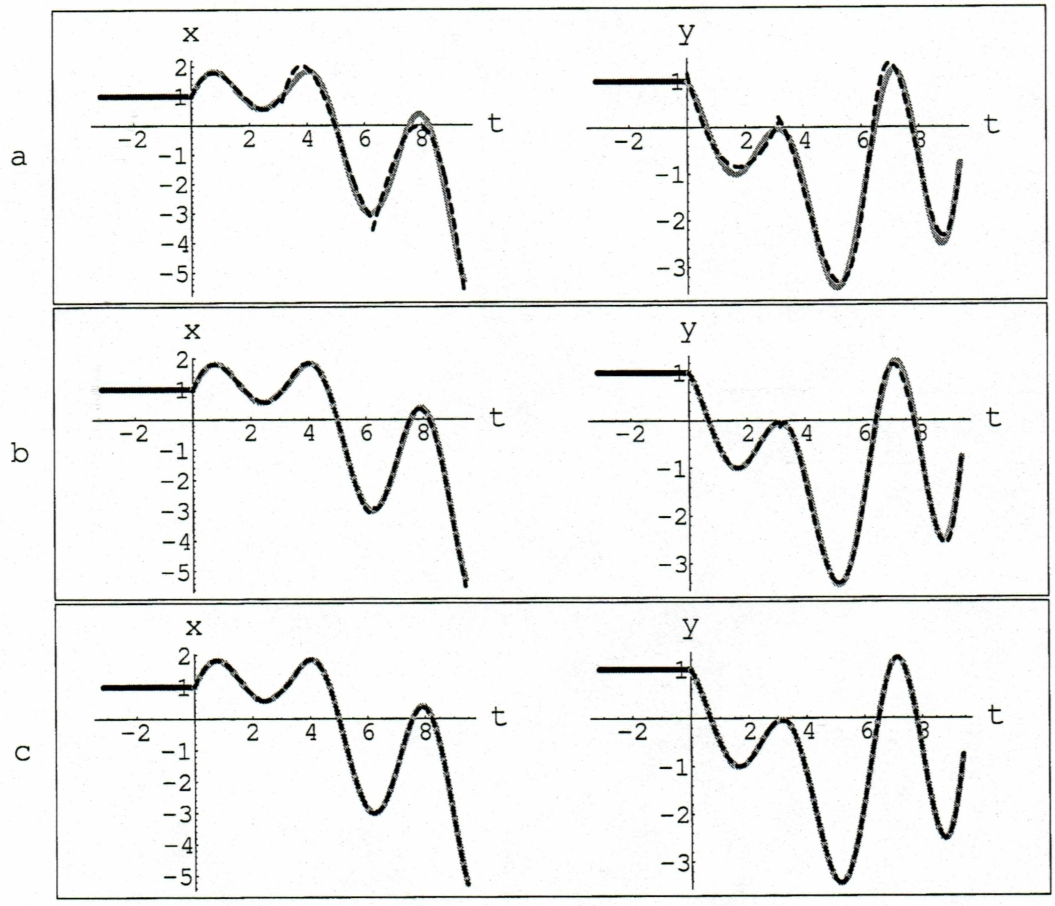

Figure 4.4: Initial condition (solid black), exact (solid grey) and approximate (dashed) solutions to (4.1) for $m = 12$, $p = 16$, and $l = (a)$ 4; (b) 5; (c) 6.

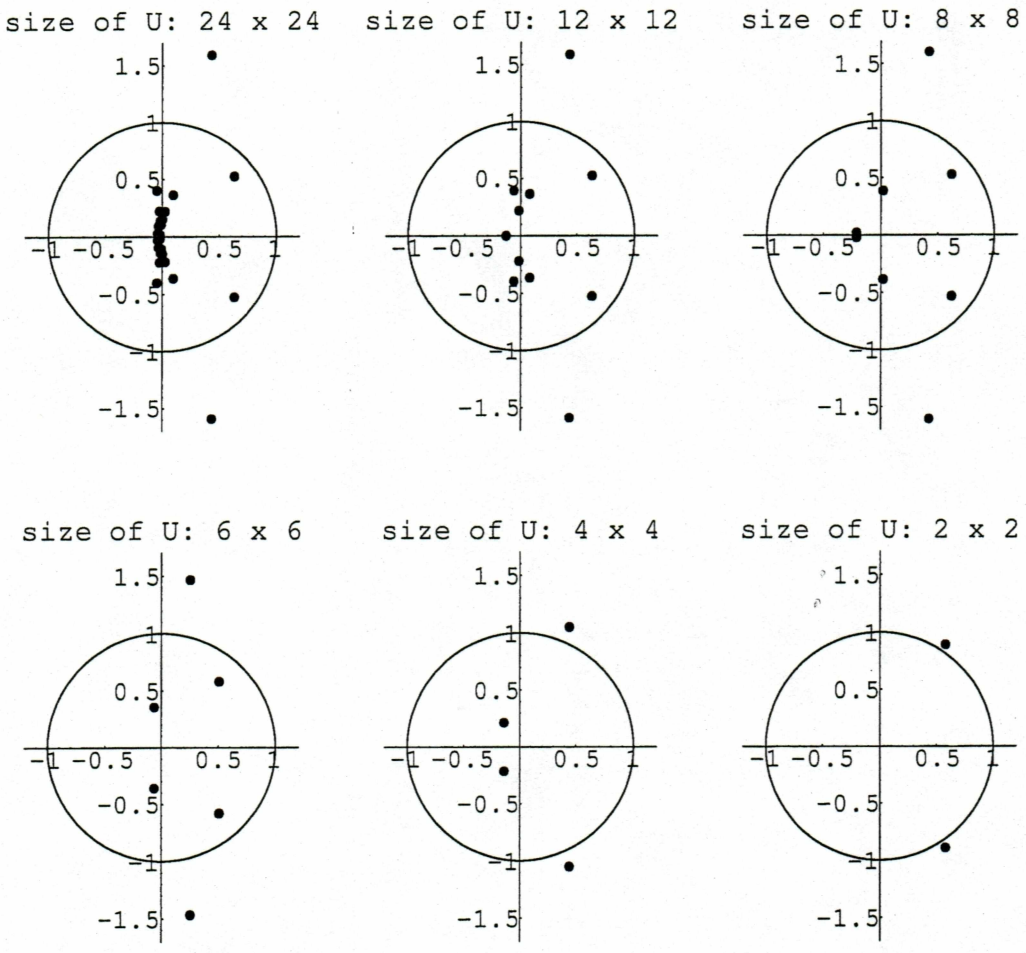

Figure 4.5: The behavior of eigenvalues of W as its size decreases.

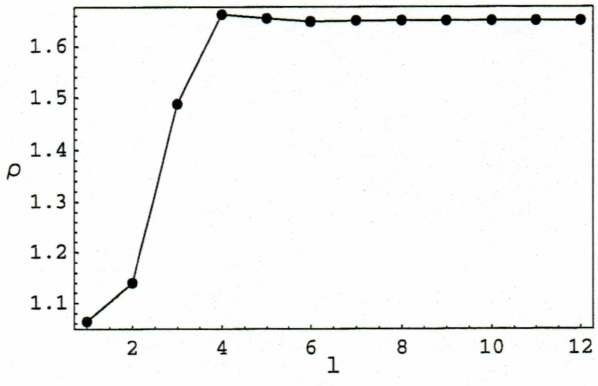

Figure 4.6: Change of the spectral radius of W with respect to its size.

4.4 Scalar constant coefficients delay differential equation

Now let us consider general scalar constant delay differential equation

$$\dot{x}(t) + ax(t) = bx(t-1). \tag{4.2}$$

Recall that we have already obtained the exact stability boundaries in (1.4) and (1.5) (see Figure 1.1). Now we want to compare them to those we get by applying Routh-Hurwitz method to $\mathbf{W}$. Let us consider region $[-1.5, 1] \times [-2.5, 1]$ of the $(a,\ b)$ parameter plane. Then according to Lemma 2 we get: $\sum_{q=p+1}^{\infty} \frac{|a|^q}{q!} \leq \sum_{q=p+1}^{\infty} \frac{1.5^q}{q!} \leq 10^{-6} \Rightarrow$ $p \geq 11$. By Proposition 2 we obtain: $\frac{\max|(e^{at})^{(m)}|}{2^{2m-1} \cdot m!} \leq \frac{1.5^m \cdot e^{-1.5}}{2^{2m-1} \cdot m!} \leq 10^{-6} \Rightarrow m \geq 7$. So let $p = 11$, $m = 7$, and $l = 7$. See Figure 4.7 for the results. The symbolic expressions for the boundaries are given in Appendix C.

4.5 Constant coefficient Mathieu equation

Consider second order Mathieu equation with constant coefficients:

$$\ddot{x}(t) + ax(t) = bx(t-\pi). \tag{4.3}$$

The analytical expressions for the stability boundaries were obtained by Insperger and Stepan in [IS].

We would want to plot and compare our approximate stability boundaries in the region $[0, 16] \times [-3.5, 2.5]$ of the $(a,\ b)$ parameter plane. However, we will need 18 polynomials and 44 Picard iterations to do so. Even if we perform analysis of the change in eigenvalues similar to the one in 4.3, we will see that $\mathbf{W}$ can be reduced from 36×36 to 22×22 which is still impossible to handle symbolically at the moment. What we will do instead is we will plot symbolic boundaries for a smaller region

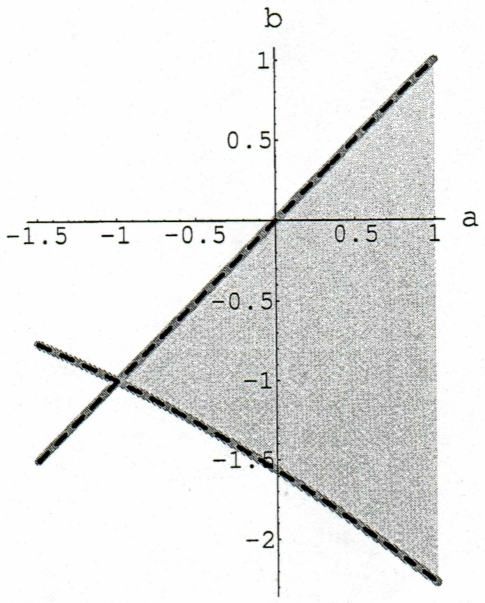

Figure 4.7: Symbolically obtained stability region (shaded area) with exact (solid grey) and approximate (dashed) boundaries for $\dot{x}(t) = ax(t) + bx(t-1)$ with $m = 7$, $l = 7$, and $p = 11$.

$[0,1] \times [0,1]$ (see Figure 4.8) and numerical boundaries for the desired region (see Figure 4.9).

4.6 Mathieu time-periodic coefficients delay differential equation

Consider second order Mathieu equation with time-periodic coefficients:

$$\ddot{x}(t) + (a + \cos t)x(t) = bx(t - 2\pi). \tag{4.4}$$

As for the previous example the analytical expressions for the stability boundaries are given in [IS]. To create a stability diagram for the region $[-1,4] \times [-1,0.5]$ of the $(a,\ b)$ parameter plane we need at least $m = 19$ and $p = 48$ and therefore we will

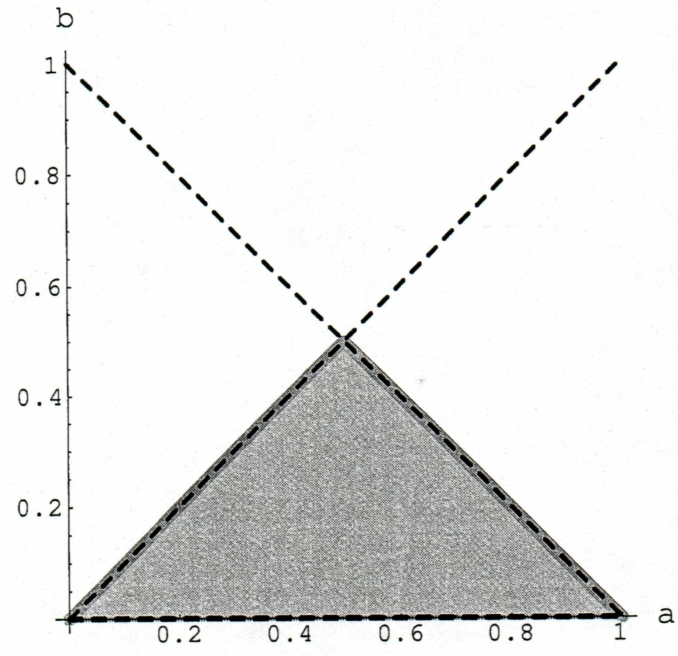

Figure 4.8: Symbolically obtained stability region (shaded area) with exact (solid grey) and approximate (dashed) boundaries for $\ddot{x}(t) + ax(t) = bx(t - \pi)$ with $m = 9$, $l = 6$, and $p = 16$.

create a numerical diagram (see Figure 4.10). The symbolic stability boundaries are impossible to obtain at this time even for a very small region.

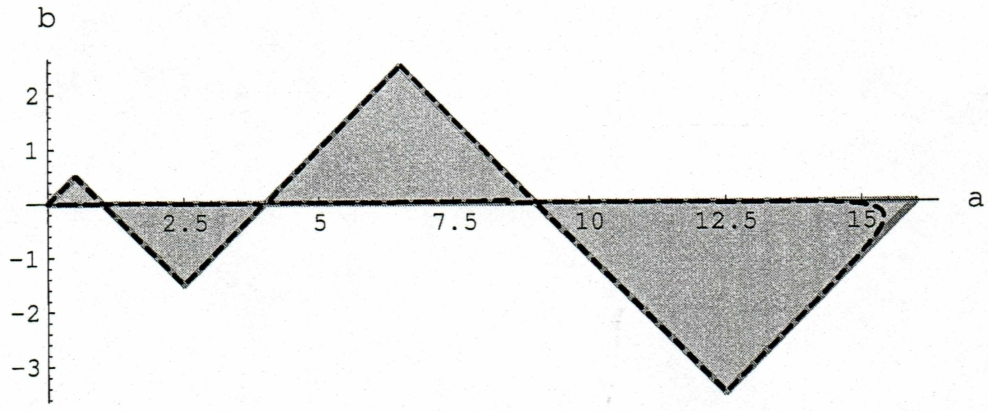

Figure 4.9: Numerically obtained stability region (shaded area) with exact (solid grey) and approximate (dashed) boundaries for $\ddot{x}(t) + ax(t) = bx(t-\pi)$ with $m = 18$, $l = 18$, $p = 44$ and step size $h = 0.01$.

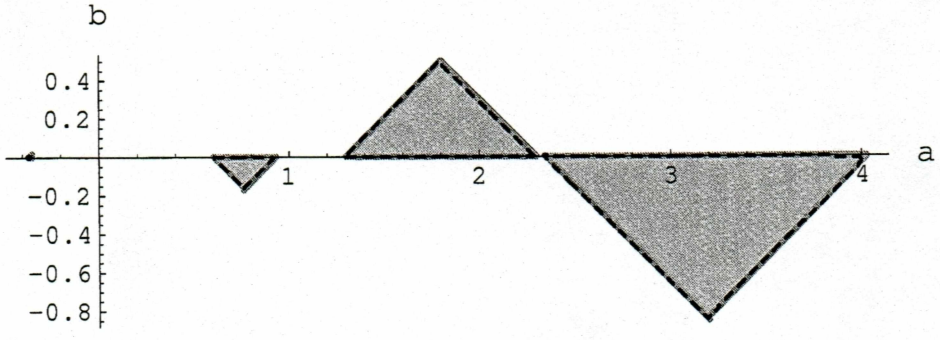

Figure 4.10: Numerically obtained stability region (shaded area) with analytical (solid grey) and approximate (dashed) boundaries for $\ddot{x}(t) + (a + \cos t)x(t) = bx(t - 2\pi)$ with $m = 19$, $l = 19$, $p = 48$ and step size $h = 0.01$.

Conclusion

In this work we have presented a new method for symbolic computation of stability boundaries of a linear system of time-periodic DDEs. Even though we have restricted ourselves here to the case when period of the coefficients is equal to the delay, this method can be extended to the more general case. There is rather big demand for such a method in engineering, in particular in high-speed machining.

We have considered both theoretical and practical matters of the topic and showed that they produce meaningful and useful results. We have also discussed the computational problems of the method and ideas for their solution.

This work is only the first step in ongoing research of symbolic computations for time-periodic DDEs. It presents a foundation for further investigation on the topic, such as bifurcation analysis.

References

[B] S. Barnett, *Polynomials and Linear Control Systems*, Marcel Dekker, Inc., 1983.

[Bu] T. Burton, *Stability and periodic solutions of ordinary and functional differential equations.* Academic Press, 1985.

[BBA] E. Butcher, E. Bueler, V. Averina, Stability of periodic linear delay-differential equations and the Chebyshev approximation of fundamental solutions. *To appear.*

[BDP] W. Boyce & R. DiPrima, *Elementary Differential Equations*, John Wiley & Sons, 2000.

[BF] R. L. Burden & D. Faires, *Numerical Analysis*, Brooks/Cole, 1997.

[BS] E.A. Butcher & S. C. Sinha, Symbolic computation of local stability and bifurcation surfaces for nonlinear time–periodic dynamical systems, *Nonlinear Dynamics*, 17, 1–21, 1998.

[CRC] *CRC Standard Mathematical Tables and Formulae*, CRC Press, 2002.

[EN] L. El'sgol'ts, S. Norkin, *Introduction to the theory and application of differential equations with deviating arguments*, Academic Press, 1973.

[HL] J. K. Hale & S. M. Verduyn Lunel, *Introduction to Functional Differential Equations*, Springer Verlag, 1993.

[I] A. Iserles, Expansions That Grow on Trees, *Notices of Amer. Math. Society*, 49, 4, 2002.

[IS] T. Insperger, G. Stepan, Stability chart for the delayed Mathieu equation, accepted in *Proceedingas of the Royal Society* (2001).

[M] W. Magnus, On the Exponential Solution of Differential Equations for a Linear Operator, *Communications on pure and applied mathematics*, VII, 649–673, 1954.

[N] Silviu-Iulian Niculescu, *Delay Effects on Stability: a robust control approach*, Springer Verlag, 2001.

[NR] W. Press, et.al., *Numerical Recipes in C*, Cambridge, 1992.

[RS] Michael Reed & Barry Simon *Functional Analysis*, Academic Press, 1980.

[R] T. J. Rivlin, *Chebyshev polynomials*, John Wiley & Sons, 1990.

[SB] S. C. Sinha & E. A. Butcher, Symbolic computation of fundamental solution matrices for linear time–periodic dynamical systems, *J. of Sound and Vibration*, 206, 1, 61–85, 1997.

[SW] S. C. Sinha & D.-H. Wu, An efficient computational scheme for the analysis of periodic systems, *J. of Sound and Vibration*, 151, 1, 91–117, 1991.

[Sz] Zsolt Szabó, Stability analysis of delayed 2nd order ODEs based on the numerical method of Chebyshev polynomials, *Preprint* (2001).

Appendix A

Initializations.nb

This notebook collects all of the *Mathematica* packages and user defined functions that are necessary to use in the notebook "DDEstab.nb".

First, we clear all the variables.

```
Apply[ClearAll, Names["Global`*"]]
```

Load all packages that will be needed.

```
<< LinearAlgebra`MatrixManipulation`
<< Algebra`InequalitySolve`
<< Graphics`ImplicitPlot`
<< Graphics`InequalityGraphics`
<< Algebra`AlgebraicInequalities`
<< DiscreteMath`Combinatorica`
<< Graphics`Animation`
```

Turn off messages about spelling errors.

```
Off[General::spell];
Off[General::spell1];
```

Function **ChebCoef[func_]** - creates a rule for Chebyshev coefficients for a given function.

```
ChebCoef[func_, m_] := Block[{x, f},
  x[k_] := .5 * Cos[Pi * (k - 0.5) / m] + 0.5;
  Table[If[j == 0,
```

```
p[j] → Expand[1/m * Sum[(func /. y -> x[k]) * ChebyshevT[0, 2 * x[k] - 1], {k, 1, m}],
p[j] → Expand[2/m * Sum[(func /. y -> x[k]) * ChebyshevT[j, 2 * x[k] - 1], {k, 1, m}]], {j,
0, m - 1}]]
```

Function **MatCoef[X_]** - creates a matrix of Chebyshev coefficients for a given time-periodic matrix.

```
MatCoef[X_] := Block[
  {temp, te, Xtemp, Y},
  temp = Table[0, {i, n}, {j, n}];
  te = Table[p[i - 1], {i, m}, {j, 1}];
  Do[Do[temp[[i, j]] = te /. ChebCoef[X[[i, j]], m], {j, 1, n}], {i, 1, n}];
  Do[Xtemp[i] = temp[[i, 1]];
   Do[Xtemp[i] = AppendRows[Xtemp[i], temp[[i, j]]], {j, 2, n}], {i, 1, n}];
  Y = Xtemp[1];
  Do[Y = AppendColumns[Y, Xtemp[i]], {i, 2, n}];
  Y]
```

Function **QMatConst[X_]** - creates product operational matrix Q for a given matrix of coefficients.

```
QMatConst[X_] := Block[
  {rule, Qtemp, Y},
  rule = ZeroMatrix[n];
  Do[Do[rule[[i, j]] = Table[p[k - 1] → X[[(i - 1) * m + k, j]], {k, 1, m}],
    {i, 1, n}], {j, 1, n}];
  Do[Qtemp[i] = Expand[Qmat /. rule[[i, 1]]];
   Do[Qtemp[i] = AppendRows[Qtemp[i], Expand[Qmat /. rule[[i, j]]]],
    {j, 2, n}], {i, 1, n}];
  Y = Qtemp[1];
  Do[Y = AppendColumns[Y, Qtemp[i]], {i, 2, n}];
  Y]
```

Function **MatCircle[X_]** - creates X°.

```
MatCircle[X_] := Block[
  {rtemp, rule, Xtemp, Y},
  temp = ZeroMatrix[n];
  rule = ZeroMatrix[n];
  Do[Do[rule[[i, j]] = Table[p[k - 1] → X[[(i - 1) * m + k, j]], {k, 1, m}],
    {i, 1, n}], {j, 1, n}];
  te = Table[p[i - 1], {j, 1}, {i, m}];
```

```
Do[Do[temp[[i, j]] = te /. rule[[i, j]], {j, 1, n}], {i, 1, n}];
Do[Xtemp[i] = temp[[i, 1]];
   Do[Xtemp[i] = AppendRows[Xtemp[i], temp[[i, j]]], {j, 2, n}], {i, 1, n}];
Y = Xtemp[1];
Do[Y = AppendColumns[Y, Xtemp[i]], {i, 2, n}];
Y = Transpose[Y];
  Y]
```

Function **CutU[l_]** - cuts nl×nl submatrix out of U.

```
CutU[l_] := Block[
  {Y, temp},
  If[l > m,
   Print["ERROR: l should be smaller or equal to m"],
   If[l == m,
    Y = U,
    Do[temp[i] = SubMatrix[U, {(i - 1) * m + 1, 1}, {l, 1}], {i, 1, n}];
    Do[Do[temp[i] = AppendRows[temp[i], SubMatrix[U,
          {(i - 1) * m + 1, (j - 1) * m + 1}, {l, 1}]], {j, 2, n}], {i, 1, n}];
    Y = temp[1];
    Do[Y = AppendColumns[Y, temp[i]], {i, 2, n}];]];
  Y]
```

Function **GM[polydeg_]** - create a **gamma matrix** to switch to Routh-Hurwitz from a characteristic polynomial with a degree **polydeg**.

```
GM[polydeg_] := Block[
  {size, Y},
  size = polydeg + 1;
  Y = ZeroMatrix[size, size];
  Do[Y[[i, size]] = 1;
   If[EvenQ[size], Y[[1, i]] = (-1) ^ (i) * Binomial[polydeg, i - 1],
    Y[[1, i]] = (-1) ^ (i + 1) * Binomial[polydeg, i - 1]],
   {i, 1, size}];
  Do[Do[Y[[i, size - j + 1]] = Y[[i - 1, size - j + 1]] + Y[[i - 1, size - j + 2]] +
      Y[[i, size - j + 2]], {j, 2, size}], {i, 2, size}];
  Y =
   N[
    Y]]
```

Function **MatrixTrace[mat_,order_]** - calculates trace of order **order** for given matrix **mat**.

```
MatrixTrace[mat_, order_] := Block[{sum, i, intersect, r, aux, time},
  (*Print["Starting ",order," trace"];*)
  time = TimeUsed[];
  sum = 0;
```

```
intersect = KSubsets[Range[Length[mat]], order];
r = Length[intersect];
Do[
 aux = intersect[[i]];
 sum = sum + Chop[N[Det[mat[[aux, aux]]]], err],
 {i, 1, r}];
Print["Trace of order ", order, ". Time used: ", TimeUsed[] - time];
sum]
```

Define a function **coef[i_]** for coefficients of characteristic polynomial for given matrix **mat.**

```
coef[i_] := Which[i < 0 || i > Length[W], 0, i == 0, 1, i == Length[W],
   Expand[(-1)^Length[W] * detw], True, (-1)^i * MatrixTrace[W, i]];
```

Function **PlotBySteps[steps_,l_]** - plots the solution for the first **steps** steps using nl×nl W matrix.

```
PlotBySteps[k_, l_] := Block[{v, te, aux, gr, W},
  W = CutU[l];
  nl = Length[W];
  timelab = StyleForm["t", FontSize → 12];
  xlab = StyleForm["x", FontSize → 12];
  ylab = StyleForm["y", FontSize → 12];
  Res[y_] = SubMatrix[Table[Expand[ChebyshevT[i, 2 * y - 1]],
     {i, 0, m - 1}, {j, 1, 1}], {1, 1}, {1, 1}];
  Reshat[y_] = N[Partition[Flatten[Outer[Times,
       IdentityMatrix[n], Res[y]]], nl]];
  te = Table[p[i - 1], {i, m}, {j, 1}];
  aux = te /. ChebCoef[delayfunc[τ * (y - 1)][[1, 1]], m];
  Do[aux = AppendColumns[aux,
     te /. ChebCoef[delayfunc[τ * (y - 1)][[i, 1]], m]], {i, 2, n}];
  v[0] = AppendColumns[SubMatrix[aux, {1, 1}, {1, 1}],
    SubMatrix[aux, {m + 1, 1}, {1, 1}]];
  Do[v[i] = W.v[i - 1], {i, 1, k}];
  func[t_] := Table[Which[t / T ≥ (j - 1) && t / T ≤ j,
     (Reshat[t / T - j + 1].v[j])[[i, 1]], True, 0], {j, k}, {i, n}];
  gr = Table[0, {i, 1, k}];
  Do[
   Do[
    gr[[i]] = Plot[func[t][[i, j]], {t, (i - 1) * T, i * T}, PlotStyle →
       {Dashing[{0.02, 0.02}], Thickness[0.01], RGBColor[0, 0, 1]},
      PlotRange → All, DisplayFunction → Identity], {i, 1, k}];
   If[j == 1, funclab = xlab, If[j == 2, funclab = ylab, funclab = ""]];
   Show[gr, AspectRatio → Automatic, DisplayFunction → $DisplayFunction,
    AxesLabel → {timelab, funclab}], {j, 1, n}];]
```

Function **RouthHurwitz[coef_,astart_,aend_,bstart_,bend_]** - plots the boundaries of the stable region for given coefficients of characteristic polynomial (with two parameters).

```
RouthHurwitz[coef_, astart_, aend_, bstart_, bend_] := Block[
  {RH, c, k, gr, time, grlen, auxx, aux},
  aux = n1 + 1;
  c[i_] := 0;
  Do[c[i] = coef[[1, aux - i]], {i, 0, aux - 1}];
  time = TimeUsed[];
  RH = Table[c[2 * j - i], {i, 1, aux - 1}, {j, 1, aux - 1}];
  Print["Created RH matrix. Time used: ", TimeUsed[] - time];
  gr = Table[0, {i, 1, aux}];
  time = TimeUsed[];
  gr[[1]] = ImplicitPlot[Chop[Expand[c[0]], err] == 0, {a, astart, aend},
    {b, bstart, bend}, PlotStyle → {Hue[1]}, DisplayFunction → Identity];
  Print["Plotted leading coefficient. Time used: ", TimeUsed[] - time];
  grlen = 1;
  Do[
   time = TimeUsed[];
   temp = TakeMatrix[RH, {1, 1}, {k, k}];
   Print["Cut ", k, " x ", k,
    " RH matrix. Time used: ", TimeUsed[] - time];
   time = TimeUsed[];
   auxx = Chop[Expand[Det[temp]], err];
   grlen = grlen + 1;
   gr[[grlen]] =
    ImplicitPlot[auxx == 0, {a, astart, aend}, {b, bstart, bend},
     PlotStyle -> Hue[k / aux], DisplayFunction → Identity];
   If[(auxx = 0) == False, grlen = grlen - 1];
   Print["Plotted ", k,
    " determinant. Time used: ", TimeUsed[] - time], {k, 1, aux - 1}];
  time = TimeUsed[];
  gr[[aux]] = ImplicitPlot[
    Chop[Expand[Det[RH]], err] == 0, {a, astart, aend}, {b, bstart, bend},
    PlotStyle → {Hue[n1 / aux]}, DisplayFunction → Identity];
  gr = Drop[gr, n1 + 1 - grlen];
  Show[gr, DisplayFunction → $DisplayFunction];
  Print["Plotted altogether. Time used: ", TimeUsed[] - time];]
```

Function **SchurCohn[coef_,astart_,aend_,bstart_,bend_]** - plots the boundaries of the stable region for given coefficients of characteristic polynomial (with two parameters).

```
SchurCohn[coef_, astart_, aend_, bstart_, bend_] := Block[
  {SC, k, gr, time, Δ1, Δ2, temp, grlen, aux},
```

```
time = TimeUsed[];
Δ1 = Table[If[i ≤ j, coef[[1, j - i + 1]], 0], {i, 1, nl}, {j, 1, nl}];
Δ2 = Table[
  If[i ≥ nl + 1 - j, coef[[1, 2 * nl + 2 - i - j]], 0], {i, 1, nl}, {j, 1, nl}];
temp = AppendRows[Δ1, Δ2];
SC = AppendRows[Δ2, Δ1];
SC = AppendColumns[temp, SC];
Print["Created SC matrix. Time used: ", TimeUsed[] - time];
gr = Table[0, {i, 1, nl}];
grlen = 1;
Do[
 time = TimeUsed[];
 temp = SC[[Range[nl - k + 1, nl + k], Range[nl - k + 1, nl + k]]];
 Print["Cut ", 2 * k, " x ", 2 * k,
  " SC matrix. Time used: ", TimeUsed[] - time];
 time = TimeUsed[];
 aux = Chop[Expand[Det[temp]], err];
 gr[[grlen]] = ImplicitPlot[aux == 0, {a, astart, aend}, {b, bstart, bend},
   PlotStyle → Hue[k / nl], DisplayFunction → Identity];
 grlen = grlen + 1;
 If[(aux == 0) == False, grlen = grlen - 1];
 Print["Plotted ", k,
  " determinant. Time used: ", TimeUsed[] - time], {k, 1, nl}];
time = TimeUsed[];
gr = Drop[gr, grlen - nl - 1];
Show[gr, DisplayFunction → $DisplayFunction];
Print["Plotted altogether. Time used: ", TimeUsed[] - time]
]
```

Function **JuryMarden[coef_,astart_,aend_,bstart_,bend_]** - plots the boundaries of the stable region for given coefficients of characteristic polynomial (with two parameters).

```
JuryMarden[coef_, astart_, aend_, bstart_, bend_] := Block[
  {JM, k, gr, aux, time, grlen},
  JM = coef;
  Print[Dimensions[JM]];
  k = 2;
  aux = Table[0, {i, 1, 1}, {j, 1, nl}];
  Print[Dimensions[aux]];
  gr = Table[0, {i, 1, nl}];
  grlen = 0;
  While[(k ≤ nl + 1),
   Print["k = ", k];
   time = TimeUsed[];
   Do[aux[[1, j]] =
```

```
    Chop[Expand[Det[( JM[[1, 1]]          JM[[1, j + 1]]
                    ( JM[[1, nl - k + 3]]  JM[[1, nl - j - k + 3]] )]], err], {j,
    1, nl + 2 - k}];
   Print["Created JM vector. Time used: ", TimeUsed[] - time];
   JM = aux;
   time = TimeUsed[];
   grlen = grlen + 1;
   gr[[grlen]] = ImplicitPlot[aux[[1, nl + 2 - k]] == 0, {a, astart, aend}, {b,
      bstart, bend}, PlotStyle -> Hue[k / nl], DisplayFunction → Identity];
   If[(aux[[1, nl + 2 - k]] == 0) == False, grlen = grlen - 1];
   Print["Plotted ", k - 1,
    " coefficient. Time used: ", TimeUsed[] - time];
   k = k + 1];
  time = TimeUsed[];
  gr = Drop[gr, grlen - nl];
  Show[gr, DisplayFunction → $DisplayFunction];
  Print["Plotted altogether. Time used: ", TimeUsed[] - time];
 ]
```

Function **Schur-Cohn-Fujiwara[coef_,astart_,aend_,bstart_,bend_]** - plots the boundaries of the stable region for given coefficients of characteristic polynomial (with two parameters).

```
SCF[coef_, astart_, aend_, bstart_, bend_] :=
 Block[{kfunc, K, gr, grlen, aux},

  kfunc[i_, j_] := Expand[∑_{r=0}^{i-1} (Chop[Expand[
          coef[[1, nl + 1 - (i - 1 - r)]] * coef[[1, nl + 1 - (j - 1 - r)]]], err] -
       Chop[Expand[coef[[1, -(r - i)]] * coef[[1, -(r - j)]]], err])];
  gr = Table[0, {i, 1, nl}];
  grlen = 0;
  Do[
   K = Table[If[i ≤ j, kfunc[i, j], kfunc[j, i]], {i, 1, q}, {j, 1, q}];
   grlen = grlen + 1;
   aux = Chop[Expand[Det[K]], err];
   gr[[grlen]] = ImplicitPlot[aux == 0, {a, astart, aend}, {b, bstart, bend},
     PlotStyle → Hue[q / nl], DisplayFunction → Identity];
   If[(aux == 0) == False, grlen = grlen - 1], {q, 1, nl}];
  gr = Drop[gr, grlen - nl];
  Show[gr, DisplayFunction → $DisplayFunction];]
```

Function **SymbStabTwo[error_]** - plots the symbolic stability boundaries of the system in the given region of (a, b)-plane with a given chopping error **error**.

```
SymbStabTwo[error_] := Block[{err},
```

```
err = error;
Print["Number of Chebyshev polynomials = ", m];
Print["Number of iterations = ", iter];
Print["chop error ", N[err]];
If[1 = m, nl = mn; W = Chop[Expand[U], err],
 W = Chop[Expand[CutU[1]], err]; nl = 1 * n];
Print["size of U: ", nl, " x ", nl];
time = TimeUsed[];
detw = Chop[Expand[Det[W]], err];
Print["Found the determinant of W. Time used: ", TimeUsed[] - time];
cvector = Table[coef[nl - j + 1], {i, 1, 1}, {j, 1, nl + 1}];
If[criterion = 1,
 Print["Routh-Hurwitz criterion"];
 pvector = N[Chop[Expand[cvector.GM[nl]], err]];
 StylePrint["Leading coefficient", FontColor → Hue[1]];
 Do[color = N[i / (nl + 1), 1]; text = ToString[i] <> " determinant";
  StylePrint[text, FontColor → Hue[color]], {i, 1, nl}];
 time = TimeUsed[];
 RouthHurwitz[pvector, astart, aend, bstart, bend];
 Print["Time used on checking the criteria: ", TimeUsed[] - time],
 If[criterion = 2, Print["Schur-Cohn criterion"];
  Do[color = N[i / nl, 1]; text = ToString[i] <> " determinant";
   StylePrint[text, FontColor → Hue[color]], {i, 1, nl}];
  SchurCohn[Reverse[cvector], astart, aend, bstart, bend], If[
   criterion = 3, Print["Jury-Marden criterion"]; Do[color = N[i / nl, 1];
    text = "c(" <> ToString[i + 1] <> ", " <> ToString[nl - i + 1] <> ")";
    StylePrint[text, FontColor → Hue[color]], {i, 1, nl}];
   JuryMarden[Reverse[cvector], astart, aend, bstart, bend],
   Print["Schur-Cohn-Fujiwara criterion"];
   Do[color = N[i / nl, 1]; text = ToString[i] <> " SCF determinant";
    StylePrint[text, FontColor → Hue[color]], {i, 1, nl}];
   SCF[Reverse[cvector], astart, aend, bstart, bend]]]]
]
```

Function **NumStabTwo[h_]** - plots the numerical stability boundaries of the system in the given region of (a, b)-plane, h - stepsize.

```
NumStabTwo[h_] := Block[{},
  Print["Period = ", T];
Print["Number of Chebyshev polynomials = ", m];
Print["Number of iterations = ", iter];
asub = Round[(aend - astart)/h];
bsub = Round[(bend - bstart)/h];
Print["Number of points used: ", (asub + 1) * (bsub + 1)];
  Ufunc[a_, b_] = U;
```

```
Uchart = ZeroMatrix[bsub + 1, asub + 1 ];
cr = 1;
Do[Do[
    ρ = N[Max[Abs[Eigenvalues[N[Ufunc[j * h + astart, i * h + bstart]]]]]];
    Print["cr = ", cr];
    If[ρ ≤ 1.001, Uchart[[i + 1, j + 1]] = 1]; cr = cr + 1,
    {j, 0, asub}],
   {i, 0, bsub}];
ListContourPlot[Uchart, Contours → 1, ContourShading → False,
   MeshRange → {{astart, aend}, {bstart, bend}}];]
```

Function **RouthHurwitzOne[coef_]** - finds the stable region **stabreg** for given coefficients **coef** of characteristic polynomial (with one parameter).

```
RouthHurwitzOne[coef_] := Block[
  {stabreg, RH, c, flag, k, aux},
  aux = nl + 1;
  c[i_] := 0;
  Do[c[i] = coef[[1, aux - i]], {i, 0, aux - 1}];
  RH = Table[c[2 * j - i], {i, 1, aux - 1}, {j, 1, aux - 1}];
  sol[0] = InequalitySolve[c[0] > 0, a];
  flag = True;
  If[sol[0] == False, flag = False];
  k = 1;
  While[(k ≤ aux - 1 && flag),
   temp = TakeMatrix[RH, {1, 1}, {k, k}];
   det[k] = Det[temp];
   sol[k] = N[InequalitySolve[sol[k - 1] && det[k] > 0, a]];
   If[sol[k] == False, flag = False];
   Print["det[", k, "] > 0. Solution: ", sol[k]];
   k = k + 1];
  Print["Stability region: ", sol[k - 1]];
 ]
```

Function **SymbStabOne** - finds symbolically the stability intervals of the system.

```
SymbStabOne := Block[{},
  Print["Number of Chebyshev polynomials = ", m];
  Print["Number of iterations = ", iter];
  Print["chop error ", N[err]];
  If[1 == m, nl = mn; W = Chop[Expand[U], err],
   W = Chop[Expand[CutU[1]], err]; nl = 1 * n];
  Print["size of U: ", nl, " x ", nl];
  time = TimeUsed[];
  detw = Chop[Expand[Det[W]], err];
  Print["Found the determinant of W. Time used: ", TimeUsed[] - time];
```

```
cvector = Table[coef[nl - j + 1], {i, 1, 1}, {j, 1, nl + 1}];
Print["Routh-Hurwitz criteria"];
pvector = N[Chop[Expand[cvector.GM[nl]], err]];
RouthHurwitzOne[pvector];]
```

Function **EigsBeh[list_,x_,y_]** - plots the eigenvalue behaviour (**part** pictures in a row) and change of the spectral radius for the sizes of approximation matrix for DFTM in **list** in the region **[-x,x] x [-y,y]**.

```
EigsBeh[list_, x_, y_, part_] := Block[{},
  grl = list;
  grlen = Length[grl];
  gr = Table[0, {i, 1, grlen}];
  uc = Plot[√(1 - t^2), {t, -1, 1}, DisplayFunction → Identity];
  lc = Plot[-√(1 - t^2), {t, -1, 1}, DisplayFunction → Identity];
  maxeigs = {};
  Do[
   W = CutU[grl[[i]]];
   temp = Eigenvalues[W];
   maxeigs = Append[maxeigs, Max[Abs[temp]]];
   aux = Table[0, {i, 1, Length[temp]}, {j, 1, 2}];
   text = StyleForm["size of U: " <> ToString[Length[temp]] <>
      " x " <> ToString[Length[temp]], FontSize → 11];
   Do[If[temp[[j]] ∈ Reals, aux[[j, 1]] = temp[[j]];
     aux[[j, 2]] = 0, aux[[j, 1]] = Re[temp[[j]]];
     aux[[j, 2]] = Im[temp[[j]]]], {j, 1, Length[temp]}];
   tempgr = ListPlot[aux, PlotStyle → PointSize[0.035], PlotRange →
      {{-x, x}, {-y, y}}, PlotLabel → text, DisplayFunction → Identity];
   gr[[i]] = Show[{tempgr, uc, lc}, AspectRatio → Automatic],
   {i, 1, grlen}];
  graph = Show[GraphicsArray[Partition[gr, part],
     AspectRatio → Automatic, DisplayFunction → $DisplayFunction,
     GraphicsSpacing → .2], ImageSize → 400];
  pl = {list, maxeigs};
 grr1 = ListPlot[Transpose[pl], PlotStyle → PointSize[0.02],
    PlotRange → All, AxesLabel → {"l", "ρ"}, DisplayFunction → Identity];
 grr2 = ListPlot[Transpose[pl], PlotStyle → PointSize[0.02],
    PlotJoined → True, DisplayFunction → Identity];
 Show[{grr1, grr2}, Axes → False, Frame → True, FrameLabel →
    {StyleForm["l", FontSize → 12], StyleForm["ρ", FontSize → 12]},
   RotateLabel → False, FrameTicks → {Automatic, Automatic, None, None},
   ImageSize → 400, DisplayFunction → $DisplayFunction];]
```

Appendix B

DDEstab.nb

Make sure you run the notebook "Initializations.nb" before using this notebook.

We are solving a delay differential equation of the following type:
$$x'(t)=A(t)x(t)+B(t)x(t-\tau),$$
where A(t) and B(t) are continuous n x n (parameter-dependent) matrix functions of time with period T, delay τ, and T = τ.

You have the following options to choose from depending on how many parameters you intend to use:

1. **Two parameters** (denote them as **a** and **b**).
   - Option 1. Plot the symbolic stability boundaries of the system in the given region of (a, b)-plane.
   - Option 2. Plot the numerical stability boundaries of the systen in the given region of (a, b)-plane.
2. **One parameter** (denote it as **a**).
   - Option 3. Find symbolically the stability intervals of the system.
3. **No parameters**.
   - Option 4. Plot the solution of the system with a given initial condition and number of steps.
   - Option 5. Plot the eigenvalue behaviour and change of the spectral radius versus size of approximation matrix for DFTM.

Enter the number of the option you chose.

```
option = 1;
```

Now, enter all the information about the system needed for the chosen option:
n - size of the matrices A(t) and B(t) (Options 1, 2, 3, 4, 5);
A[t_] - matrix A(t) (Options 1, 2, 3, 4, 5);
B[t_] - matrix B(t) (Options 1, 2, 3, 4, 5);
T - the period of A(t) and B(t)) (**T** = τ) (Options 1, 2, 3, 4, 5);
m - number of Chebyshev polynomials you want to employ (Options 1, 2, 3, 4, 5);
iter - number of Pichard iterations (Options 1, 2, 3, 4, 5);
l - size of the block in U - for truncating U to get W (Options 1, 2, 3, 4);
[astart, aend] x [bstart, bend] - region in the (a, b)-plane (Options 1, 2);
delayfunc[t_] - initial condition function (Option 4);
steps - number of steps (Options 4);
criterion - stability criterion:

1 if Routh-Hurwitz,
2 if Schur-Cohn,
3 if Jury-Marden,
4 if Schur-Cohn-Fujiwara.
RECOMMENDED: Routh-Hurwitz.

Ignore the variables you do not need for the chosen option.
Note that **m** and **iter** can be found using Lemma 3, Proprosition 2, and "m and p estimations.nb".

```
n = 2;
a = 1;
A[t_] = ( -1 + a * Cos[t] * Cos[t]   1 - a * Sin[t] * Cos[t]
          -1 - a * Sin[t] * Cos[t]  -1 + a * Sin[t] * Sin[t] );
B[t_] = ( 1  0
          0  1 );
T = 1;
m = 3;
iter = 1;
l = m;
astart = 0;
aend = 3;
bstart = -2;
bend = 1;
delayfunc[t_] = {{1}, {1}};
steps = 3;
criterion = 1;
```

Now, evaluate this notebook (Kernel - Evaluation - Evaluate Notebook). Check if you see any error messages at the end of this program!

CHECK FOR THE ERRORS IN THE DATA! This part checks if you entered data correctly. DO NOT change it!

```
flag = 0;
If[Dimensions[A[y]] ≠ Dimensions[B[y]] || Dimensions[B[y]] ≠ {n, n},
  Print[StyleForm["ERROR", FontSize → 14, FontColor → RGBColor[1, 0, 0]],
   ": Matrices A(t) and B(t) should be of the same size: ",
   n, " x ", n, "! So either change matrices or n."]; flag = 1];
If[m < 3, Print[StyleForm["ERROR", FontSize → 14,
   FontColor → RGBColor[1, 0, 0]], ": m should be at least equal to 3."];
 flag = 1]; If[option == 4 & Dimensions[delayfunc[y]] ≠ {n, 1} ,
 Print[StyleForm["ERROR", FontSize → 14, FontColor → RGBColor[1, 0, 0]],
  ": delayfunc[y] should be a matrix of the size: ",
  n, " x ", n]; flag = 1];
If[option < 1 || option > 5,
  Print[StyleForm["ERROR", FontSize → 14, FontColor → RGBColor[1, 0, 0]],
   ": option value should be between 1 and 5."]; flag = 1];
If[flag == 0, Print["Data is correct."], Quit[]];
```

The following is the main part of the program - creating an approximation matrix U for DFTM.

Automatic definitions and matrix normalization.

```
mn = m n;
n1 = n 1;
Anorm[y_] = A[t] /. t → y * T;
Bnorm[y_] = B[t] /. t → y * T;
amat[y_] = T * Anorm[y];
bmat[y_] = T * Bnorm[y];
τ = T;
err = 10^(-1000);
```

Create integration operational matrix G (constant matrix m×m).

```
G = Table[0, {i, 1, m}, {j, 1, m}];
(* Consider first two rows separtely *)
G[[1, 1]] = 1 / 2;
G[[1, 2]] = 1 / 2;
G[[2, 1]] = -1 / 8;
G[[2, 3]] = 1 / 8;
(* Now fill in the rest of the table G *)
For[i = 3, i ≤ m,
  For[j = 1, j ≤ m,
   If[j == 1, G[[i, j]] = (-1)^(i - 1) / (2 * (1 - (i - 1)^2))];
   If[j + 1 == i, G[[i, j]] = -1 / (4 * i - 8)];
   If[j - 1 == i, G[[i, j]] = 1 / (4 * i)];
   j++]; i++];
```

Define Q matrix in terms of Chebyshev coefficients $p[i]$.

```
mult[i_, j_] := mult[i, j] = (tstar[i + j - 2] + tstar[Abs[i - j]]) / 2
Do[pmult[j] = Sum[p[i - 1] * mult[i, j], {i, 1, m}], {j, 1, m}]
Do[pmult[j] = Expand[pmult[j]], {j, 1, m}]
Do[c[j] = {}, {j, 1, m}]
Do[Do[c[j] = Append[c[j], Coefficient[pmult[j], tstar[i], 1]],
  {i, 0, m - 1}], {j, 1, m}]
Qmat = {};
Do[Qmat = Append[Qmat, c[i]], {i, 1, m}]
Qmat = Transpose[Qmat];
```

Create matrix of coefficients for amat(y) = $\hat{T}(y)'A$.

```
A = MatCoef[amat[y]];
```

Create product operational matrix for A (Q_A=QA).

```
QA = QMatConst[A];
```

Create $W_A = Q_A * G'$.

```
Gt = Transpose[G];
Gt = Outer[Times, IdentityMatrix[n], Gt];
Gprime = {};
Do[Do[Do[Gprime = Append[Gprime, Gt[[i, k, j]]], {k, 1, n}], {j, 1, m}],
 {i, 1, n}]
Gprime = Partition[Flatten[Gprime], mn];
WA = Chop[Expand[QA.Gprime], err];
```

Create "present" fundamental solution $\Phi = I + G'(A + W_A(A + \ldots + W_A(A + W_A A)\ldots)) = \hat{T}(y)'\Phi$.

Create identity coefficient matrix.

```
temp = Flatten[Append[{1}, ZeroMatrix[1, m - 1]]];
Ihat = Transpose[
   Partition[Flatten[Outer[Times, IdentityMatrix[n], temp]], mn]];

sum = Chop[Expand[N[A]], err];
For[j = 1, j ≤ iter - 1, j++, Print["working on p = ", j + 1];
  temp = Chop[Expand[N[WA.sum]], err];
  sum = Chop[N[A], err] + temp];
Print["Expanding the sum"];
sum = Chop[Expand[N[sum]], err];
Print["Working on Φ"];
Φ = Chop[Expand[N[Ihat + Gprime.sum]], err];
(* Define shat-- matrix of Cheb polynomials = T hat transposed *)
sstar = {};
```

```
Do[sstar = Append[sstar, Expand[ChebyshevT[j, 2 * t - 1]]], {j, 0, m - 1}]
Print["Working on shat"];
shat =
  N[Partition[Flatten[Outer[Times, IdentityMatrix[n], sstar]], mn]];
Print["Working on ftm"];
ftm = Chop[Expand[N[shat.Φ]], err];
```

Create $A^{\circ} = (A')^{T}$.

```
Acircle = MatCircle[A];
```

Create product operational matrix for A° ($Q_{A^{\circ}}$=QAcircle=$(Q_A)^T$).

```
QAcircle = QMatConst[Acircle];
```

Create "delay" fundamental solution Ψ=I-G'(A°-W_A(A°-...-W_A(A°-$W_A A^{\circ}$)...))=$\hat{T}(y)'\Psi$.

```
WAT = Chop[N[QAcircle.Gprime], err];
temp = sum = Chop[N[Acircle], err];
For[j = 1, j ≤ iter - 1, j++, Print["working on p = ", j + 1];
  temp = Chop[Expand[N[WAT.sum]], err];
  sum = Chop[Expand[N[Acircle - temp]], err]];
sum = Chop[Expand[N[sum]], err];
Ψ = Chop[Expand[N[Ihat - Gprime.sum]], err];
Print["Working on psiftm"];
psiftm = Chop[Expand[shat.Ψ], err];
```

Create the transition matrix U=$\Phi\hat{T}(1)'$+Q_{Φ}G'Q_Z=Φ*Tone+QΦ*Gprime*QZ where Z=Q_{Ψ^T} BB =QΨT*BB, where BB is matrix of Chebyshev coefficients for bmat.

Create $\hat{T}(1)'$=I×$T^T(1)$ =Tone.

```
Tone = Table[1, {j, 1}, {i, m}];
Tone = Partition[Flatten[Outer[Times, IdentityMatrix[n], Tone]], mn];
```

Create matrix of coefficients for bmat(y)=$\hat{T}(y)'$BB.

```
BB = MatCoef[bmat[y]];
```

Create Z= Q_{Ψ^T} BB =QΨT*BB.

```
QΨT = Chop[QMatConst[MatCircle[Ψ]], err];
Z = Chop[Expand[N[QΨT.BB]], err];
```

Create Q_Z=QZ.

```
QZ = Chop[QMatConst[Z], err];
```

Create $Q_\Phi = Q\Phi$.

```
QΦ = Chop[QMatConst[Φ], err];
```

Put everything together in U.

```
U = Φ.Tone + QΦ.Gprime.QZ;
```

The result of this part depends on the option you chose.

```
If[option == 1, SymbStabTwo[10^(-6)],
  If[option == 2, NumStabTwo[0.5], If[option == 3, SymbStabOne,
    If[option == 4, PlotBySteps[steps, 1], EigsBeh[{1, 2, 3}, 1, 1, 3]]]]];
```

Appendix C

Approximate Symbolic Stability Boundaries

The following are the approximate symbolic stability boundaries for (4.2) obtained using *Mathematica* programs shown in Appendices A and B.

The approximate stability boundary corresponding to (1.4):

$$-1.\,a - 0.5\,a^2 - 0.166667\,a^3 - 0.0416667\,a^4 - 0.00833333\,a^5 - 0.00138821\,a^6 - 0.000198049\,a^7 - 0.0000242738\,a^8 - 1.\,b - 0.5\,a\,b - 0.25\,a^2\,b - 0.0833333\,a^3\,b - 0.0215495\,a^4\,b - 0.00452338\,a^5\,b - 0.000833309\,a^6\,b - 0.000138728\,a^7\,b - 0.0000161038\,a^8\,b - 1.62669\times10^{-6}\,a^9\,b - 0.114583\,a\,b^2 - 0.0572917\,a^2\,b^2 - 0.0199436\,a^3\,b^2 - 0.00519748\,a^4\,b^2 - 0.00117374\,a^5\,b^2 - 0.000233343\,a^6\,b^2 - 0.0000405251\,a^7\,b^2 - 3.79229\times10^{-6}\,a^8\,b^2 - 0.03125\,b^3 - 0.015625\,a\,b^3 - 0.00794271\,a^2\,b^3 - 0.00266927\,a^3\,b^3 - 0.000757062\,a^4\,b^3 - 0.000174706\,a^5\,b^3 - 0.0000371549\,a^6\,b^3 - 2.76268\times10^{-6}\,a^7\,b^3 - 0.00136719\,a\,b^4 - 0.000683594\,a^2\,b^4 - 0.000263347\,a^3\,b^4 - 0.0000714655\,a^4\,b^4 - 0.0000157349\,a^5\,b^4 - 1.60489\times10^{-6}\,a^6\,b^4 - 0.00015191\,b^5 - 0.0000759549\,a\,b^5 - 0.0000431848\,a^2\,b^5 - 0.000015552\,a^3\,b^5 - 4.75904\times10^{-6}\,a^4\,b^5 - 1.24919\times10^{-6}\,a^5\,b^5 - 2.20405\times10^{-6}\,a\,b^6 - 1.08083\times10^{-6}\,a^2\,b^6 = 0.$$

The approximate stability boundary corresponding to (1.5):

$$4.1943\times10^{6}+2.09715\times10^{6}\,a+1.04858\times10^{6}\,a^{2}+349525.\,a^{3}+87381.3\,a^{4}+17476.3\,a^{5}+$$
$$2911.29\,a^{6}+415.34\,a^{7}+50.9058\,a^{8}+2.09715\times10^{6}\,b+2.09715\times10^{6}\,a\,b+2.09715\times10^{6}\,a^{2}\,b+$$
$$1.3981\times10^{6}\,a^{3}\,b+702054.\,a^{4}\,b+282621.\,a^{5}\,b+95290.\,a^{6}\,b+27712.5\,a^{7}\,b+7100.16\,a^{8}\,b+$$
$$1621.81\,a^{9}\,b+331.466\,a^{10}\,b+60.4965\,a^{11}\,b+9.81414\,a^{12}\,b+1.40547\,a^{13}\,b+0.175434\,a^{14}\,b+$$
$$0.0184954\,a^{15}\,b+0.00149541\,a^{16}\,b+0.000082808\,a^{17}\,b+480597.\,b^{2}+677205.\,a\,b^{2}+$$
$$776055.\,a^{2}\,b^{2}+792940.\,a^{3}\,b^{2}+647545.\,a^{4}\,b^{2}+419483.\,a^{5}\,b^{2}+222545.\,a^{6}\,b^{2}+99907.5\,a^{7}\,b^{2}+$$
$$38961.7\,a^{8}\,b^{2}+13468.9\,a^{9}\,b^{2}+4186.03\,a^{10}\,b^{2}+1181.13\,a^{11}\,b^{2}+304.424\,a^{12}\,b^{2}+71.9274\,a^{13}\,b^{2}+$$
$$15.605\,a^{14}\,b^{2}+3.10951\,a^{15}\,b^{2}+0.568281\,a^{16}\,b^{2}+0.0949354\,a^{17}\,b^{2}+0.0144044\,a^{18}\,b^{2}+$$
$$0.00196628\,a^{19}\,b^{2}+0.000238202\,a^{20}\,b^{2}+0.0000250801\,a^{21}\,b^{2}+2.20851\times10^{-6}\,a^{22}\,b^{2}+$$
$$677205.\,b^{3}+1.11411\times10^{6}\,a\,b^{3}+918869.\,a^{2}\,b^{3}+593033.\,a^{3}\,b^{3}+358725.\,a^{4}\,b^{3}+223244.\,a^{5}\,b^{3}+$$
$$139409.\,a^{6}\,b^{3}+81763.9\,a^{7}\,b^{3}+43273.7\,a^{8}\,b^{3}+20449.1\,a^{9}\,b^{3}+8655.93\,a^{10}\,b^{3}+3307.34\,a^{11}\,b^{3}+$$
$$1149.65\,a^{12}\,b^{3}+365.979\,a^{13}\,b^{3}+107.26\,a^{14}\,b^{3}+29.0582\,a^{15}\,b^{3}+7.29895\,a^{16}\,b^{3}+1.70341\,a^{17}\,b^{3}+$$
$$0.369788\,a^{18}\,b^{3}+0.0746913\,a^{19}\,b^{3}+0.0140277\,a^{20}\,b^{3}+0.00244564\,a^{21}\,b^{3}+0.000394734\,a^{22}\,b^{3}+$$
$$0.0000587378\,a^{23}\,b^{3}+8.00967\times10^{-6}\,a^{24}\,b^{3}+114506.\,b^{4}+216724.\,a\,b^{4}+258620.\,a^{2}\,b^{4}+$$
$$243722.\,a^{3}\,b^{4}+184235.\,a^{4}\,b^{4}+117162.\,a^{5}\,b^{4}+67008.\,a^{6}\,b^{4}+36676.7\,a^{7}\,b^{4}+19839.8\,a^{8}\,b^{4}+$$
$$10535.1\,a^{9}\,b^{4}+5352.74\,a^{10}\,b^{4}+2544.45\,a^{11}\,b^{4}+1117.56\,a^{12}\,b^{4}+451.195\,a^{13}\,b^{4}+167.085\,a^{14}\,b^{4}+$$
$$56.5867\,a^{15}\,b^{4}+17.3885\,a^{16}\,b^{4}+4.74964\,a^{17}\,b^{4}+1.09061\,a^{18}\,b^{4}+0.171306\,a^{19}\,b^{4}-$$
$$0.00900613\,a^{20}\,b^{4}-0.0234284\,a^{21}\,b^{4}-0.0134623\,a^{22}\,b^{4}-0.00584753\,a^{23}\,b^{4}-0.0022073\,a^{24}\,b^{4}-$$
$$0.000758397\,a^{25}\,b^{4}-0.000242171\,a^{26}\,b^{4}-0.0000726598\,a^{27}\,b^{4}-0.0000206145\,a^{28}\,b^{4}-$$
$$5.55232\times10^{-6}\,a^{29}\,b^{4}-1.42337\times10^{-6}\,a^{30}\,b^{4}+41961.2\,b^{5}+72544.7\,a\,b^{5}+81417.1\,a^{2}\,b^{5}+$$
$$75867.1\,a^{3}\,b^{5}+63980.2\,a^{4}\,b^{5}+48210.2\,a^{5}\,b^{5}+31833.\,a^{6}\,b^{5}+18457.9\,a^{7}\,b^{5}+9610.98\,a^{8}\,b^{5}+$$
$$4651.08\,a^{9}\,b^{5}+2174.37\,a^{10}\,b^{5}+1011.59\,a^{11}\,b^{5}+472.166\,a^{12}\,b^{5}+218.245\,a^{13}\,b^{5}+97.8146\,a^{14}\,b^{5}+$$
$$41.7827\,a^{15}\,b^{5}+16.8508\,a^{16}\,b^{5}+6.39894\,a^{17}\,b^{5}+2.29237\,a^{18}\,b^{5}+0.778481\,a^{19}\,b^{5}+0.25228\,a^{20}\,b^{5}+$$
$$0.0786169\,a^{21}\,b^{5}+0.0237437\,a^{22}\,b^{5}+0.00699886\,a^{23}\,b^{5}+0.00202385\,a^{24}\,b^{5}+0.000575534\,a^{25}\,b^{5}+$$
$$0.000160894\,a^{26}\,b^{5}+0.0000440992\,a^{27}\,b^{5}+0.0000118038\,a^{28}\,b^{5}+3.07214\times10^{-6}\,a^{29}\,b^{5}+$$
$$15841.2\,b^{6}+38525.3\,a\,b^{6}+48770.2\,a^{2}\,b^{6}+41922.6\,a^{3}\,b^{6}+28061.9\,a^{4}\,b^{6}+16041.1\,a^{5}\,b^{6}+$$
$$8469.45\,a^{6}\,b^{6}+4370.82\,a^{7}\,b^{6}+2230.31\,a^{8}\,b^{6}+1089.15\,a^{9}\,b^{6}+484.107\,a^{10}\,b^{6}+185.288\,a^{11}\,b^{6}+$$
$$55.9079\,a^{12}\,b^{6}+9.70763\,a^{13}\,b^{6}-2.25805\,a^{14}\,b^{6}-3.19914\,a^{15}\,b^{6}-1.9279\,a^{16}\,b^{6}-0.881006\,a^{17}\,b^{6}-$$

$$0.339144\,a^{18}\,b^6 - 0.114104\,a^{19}\,b^6 - 0.0340059\,a^{20}\,b^6 - 0.00896881\,a^{21}\,b^6 - 0.0020559\,a^{22}\,b^6 - 0.000387255\,a^{23}\,b^6 - 0.0000484508\,a^{24}\,b^6 + 2.35696\times10^{-6}\,a^{25}\,b^6 + 4.22475\times10^{-6}\,a^{26}\,b^6 + 1.82904\times10^{-6}\,a^{27}\,b^6 - 218.074\,b^7 + 961.6\,a\,b^7 + 2652.01\,a^2\,b^7 + 3788.23\,a^3\,b^7 + 3791.63\,a^4\,b^7 + 2877.67\,a^5\,b^7 + 1689.43\,a^6\,b^7 + 739.294\,a^7\,b^7 + 186.908\,a^8\,b^7 - 45.2543\,a^9\,b^7 - 98.4663\,a^{10}\,b^7 - 81.3631\,a^{11}\,b^7 - 50.7804\,a^{12}\,b^7 - 27.0046\,a^{13}\,b^7 - 12.8351\,a^{14}\,b^7 - 5.59489\,a^{15}\,b^7 - 2.27378\,a^{16}\,b^7 - 0.871392\,a^{17}\,b^7 - 0.317495\,a^{18}\,b^7 - 0.110638\,a^{19}\,b^7 - 0.0370351\,a^{20}\,b^7 - 0.011947\,a^{21}\,b^7 - 0.00372286\,a^{22}\,b^7 - 0.00112267\,a^{23}\,b^7 - 0.000328069\,a^{24}\,b^7 - 0.0000929882\,a^{25}\,b^7 - 0.0000255798\,a^{26}\,b^7 - 6.83106\times10^{-6}\,a^{27}\,b^7 - 1.77081\times10^{-6}\,a^{28}\,b^7 + 128.231\,b^8 + 518.301\,a\,b^8 + 742.193\,a^2\,b^8 + 696.809\,a^3\,b^8 + 502.26\,a^4\,b^8 + 292.972\,a^5\,b^8 + 133.819\,a^6\,b^8 + 39.5026\,a^7\,b^8 - 1.48713\,a^8\,b^8 - 10.6303\,a^9\,b^8 - 7.25786\,a^{10}\,b^8 - 2.40907\,a^{11}\,b^8 + 0.325357\,a^{12}\,b^8 + 1.10494\,a^{13}\,b^8 + 0.954889\,a^{14}\,b^8 + 0.592815\,a^{15}\,b^8 + 0.305366\,a^{16}\,b^8 + 0.137796\,a^{17}\,b^8 + 0.0560238\,a^{18}\,b^8 + 0.0208817\,a^{19}\,b^8 + 0.00722202\,a^{20}\,b^8 + 0.00233918\,a^{21}\,b^8 + 0.000715047\,a^{22}\,b^8 + 0.000207726\,a^{23}\,b^8 + 0.0000577325\,a^{24}\,b^8 + 0.0000154519\,a^{25}\,b^8 + 4.00843\times10^{-6}\,a^{26}\,b^8 + 1.01394\times10^{-6}\,a^{27}\,b^8 + 11.0459\,b^9 + 80.8933\,a\,b^9 + 168.216\,a^2\,b^9 + 181.275\,a^3\,b^9 + 119.968\,a^4\,b^9 + 48.4719\,a^5\,b^9 + 9.33197\,a^6\,b^9 + 1.20414\,a^7\,b^9 + 5.43827\,a^8\,b^9 + 9.06891\,a^9\,b^9 + 8.91172\,a^{10}\,b^9 + 6.55906\,a^{11}\,b^9 + 3.98263\,a^{12}\,b^9 + 2.09809\,a^{13}\,b^9 + 0.989226\,a^{14}\,b^9 + 0.426447\,a^{15}\,b^9 + 0.17075\,a^{16}\,b^9 + 0.0642692\,a^{17}\,b^9 + 0.0229535\,a^{18}\,b^9 + 0.00783504\,a^{19}\,b^9 + 0.00257026\,a^{20}\,b^9 + 0.000813598\,a^{21}\,b^9 + 0.000249199\,a^{22}\,b^9 + 0.000073982\,a^{23}\,b^9 + 0.0000213066\,a^{24}\,b^9 + 5.95385\times10^{-6}\,a^{25}\,b^9 + 1.61389\times10^{-6}\,a^{26}\,b^9 - 51.4374\,b^{10} - 135.254\,a\,b^{10} - 178.698\,a^2\,b^{10} - 163.477\,a^3\,b^{10} - 116.175\,a^4\,b^{10} - 67.8987\,a^5\,b^{10} - 33.9732\,a^6\,b^{10} - 15.3502\,a^7\,b^{10} - 6.87379\,a^8\,b^{10} - 3.42442\,a^9\,b^{10} - 1.96125\,a^{10}\,b^{10} - 1.1904\,a^{11}\,b^{10} - 0.701243\,a^{12}\,b^{10} - 0.384614\,a^{13}\,b^{10} - 0.194408\,a^{14}\,b^{10} - 0.0907637\,a^{15}\,b^{10} - 0.0393688\,a^{16}\,b^{10} - 0.0159624\,a^{17}\,b^{10} - 0.00608332\,a^{18}\,b^{10} - 0.00218948\,a^{19}\,b^{10} - 0.00074729\,a^{20}\,b^{10} - 0.000242754\,a^{21}\,b^{10} - 0.0000753033\,a^{22}\,b^{10} - 0.000022375\,a^{23}\,b^{10} - 6.38658\times10^{-6}\,a^{24}\,b^{10} - 1.75595\times10^{-6}\,a^{25}\,b^{10} - 4.25237\,b^{11} - 11.4219\,a\,b^{11} - 17.0992\,a^2\,b^{11} - 19.2143\,a^3\,b^{11} - 17.5416\,a^4\,b^{11} - 13.5511\,a^5\,b^{11} - 9.19958\,a^6\,b^{11} - 5.69335\,a^7\,b^{11} - 3.30491\,a^8\,b^{11} - 1.82697\,a^9\,b^{11} - 0.964324\,a^{10}\,b^{11} - 0.484084\,a^{11}\,b^{11} - 0.230021\,a^{12}\,b^{11} - 0.103179\,a^{13}\,b^{11} - 0.0436743\,a^{14}\,b^{11} - 0.0174676\,a^{15}\,b^{11} - 0.00661567\,a^{16}\,b^{11} - 0.00237931\,a^{17}\,b^{11} - 0.000815157\,a^{18}\,b^{11} - 0.00026698\,a^{19}\,b^{11} - 0.0000839141\,a^{20}\,b^{11} - 0.0000254135\,a^{21}\,b^{11} - 7.44547\times10^{-6}\,a^{22}\,b^{11} - 2.11775\times10^{-6}\,a^{23}\,b^{11} - 1.27617\,b^{12} - 3.11716\,a\,b^{12} - 4.01776\,a^2\,b^{12} -$$

$$3.63485\,a^3\,b^{12} - 2.62679\,a^4\,b^{12} - 1.61668\,a^5\,b^{12} - 0.844106\,a^6\,b^{12} - 0.332135\,a^7\,b^{12} -$$
$$0.0448356\,a^8\,b^{12} + 0.0759523\,a^9\,b^{12} + 0.0977869\,a^{10}\,b^{12} + 0.0774691\,a^{11}\,b^{12} + 0.0492349\,a^{12}\,b^{12} +$$
$$0.0270881\,a^{13}\,b^{12} + 0.0133666\,a^{14}\,b^{12} + 0.00603819\,a^{15}\,b^{12} + 0.00253077\,a^{16}\,b^{12} +$$
$$0.000993408\,a^{17}\,b^{12} + 0.000367721\,a^{18}\,b^{12} + 0.000129034\,a^{19}\,b^{12} + 0.0000431019\,a^{20}\,b^{12} +$$
$$0.0000137522\,a^{21}\,b^{12} + 4.20318\times10^{-6}\,a^{22}\,b^{12} + 1.23361\times10^{-6}\,a^{23}\,b^{12} - 0.212656\,b^{13} -$$
$$0.678422\,a\,b^{13} - 1.12222\,a^2\,b^{13} - 1.206\,a^3\,b^{13} - 0.917842\,a^4\,b^{13} - 0.501761\,a^5\,b^{13} - 0.177838\,a^6\,b^{13} -$$
$$0.0112016\,a^7\,b^{13} + 0.0399956\,a^8\,b^{13} + 0.0382116\,a^9\,b^{13} + 0.0235443\,a^{10}\,b^{13} + 0.0116128\,a^{11}\,b^{13} +$$
$$0.00491289\,a^{12}\,b^{13} + 0.00184245\,a^{13}\,b^{13} + 0.00062323\,a^{14}\,b^{13} + 0.000191302\,a^{15}\,b^{13} +$$
$$0.000052878\,a^{16}\,b^{13} + 0.0000127345\,a^{17}\,b^{13} + 2.40232\times10^{-6}\,a^{18}\,b^{13} + 0.0335977\,b^{14} +$$
$$0.0980465\,a\,b^{14} + 0.15588\,a^2\,b^{14} + 0.17901\,a^3\,b^{14} + 0.158343\,a^4\,b^{14} + 0.109217\,a^5\,b^{14} +$$
$$0.057697\,a^6\,b^{14} + 0.0209923\,a^7\,b^{14} + 0.00195477\,a^8\,b^{14} - 0.00468688\,a^9\,b^{14} - 0.00521309\,a^{10}\,b^{14} -$$
$$0.00376039\,a^{11}\,b^{14} - 0.00222434\,a^{12}\,b^{14} - 0.00115649\,a^{13}\,b^{14} - 0.000545303\,a^{14}\,b^{14} -$$
$$0.000237244\,a^{15}\,b^{14} - 0.0000962825\,a^{16}\,b^{14} - 0.0000367299\,a^{17}\,b^{14} - 0.0000132483\,a^{18}\,b^{14} -$$
$$4.54005\times10^{-6}\,a^{19}\,b^{14} - 1.48421\times10^{-6}\,a^{20}\,b^{14} + 0.00376233\,b^{15} + 0.00948899\,a\,b^{15} +$$
$$0.0126965\,a^2\,b^{15} + 0.0117583\,a^3\,b^{15} + 0.00802881\,a^4\,b^{15} + 0.0040148\,a^5\,b^{15} + 0.00139592\,a^6\,b^{15} +$$
$$0.000339662\,a^7\,b^{15} + 0.000176766\,a^8\,b^{15} + 0.000258935\,a^9\,b^{15} + 0.000291049\,a^{10}\,b^{15} +$$
$$0.000240303\,a^{11}\,b^{15} + 0.000159303\,a^{12}\,b^{15} + 0.0000897447\,a^{13}\,b^{15} + 0.0000445146\,a^{14}\,b^{15} +$$
$$0.0000199085\,a^{15}\,b^{15} + 8.16677\times10^{-6}\,a^{16}\,b^{15} + 3.11282\times10^{-6}\,a^{17}\,b^{15} + 1.11357\times10^{-6}\,a^{18}\,b^{15} +$$
$$0.000742957\,b^{16} + 0.00187353\,a\,b^{16} + 0.00256682\,a^2\,b^{16} + 0.00260086\,a^3\,b^{16} +$$
$$0.00226337\,a^4\,b^{16} + 0.00181042\,a^5\,b^{16} + 0.00134286\,a^6\,b^{16} + 0.000911723\,a^7\,b^{16} +$$
$$0.000563566\,a^8\,b^{16} + 0.000319982\,a^9\,b^{16} + 0.000169907\,a^{10}\,b^{16} + 0.0000859858\,a^{11}\,b^{16} +$$
$$0.0000420049\,a^{12}\,b^{16} + 0.000019873\,a^{13}\,b^{16} + 9.06721\times10^{-6}\,a^{14}\,b^{16} + 3.96086\times10^{-6}\,a^{15}\,b^{16} +$$
$$1.64615\times10^{-6}\,a^{16}\,b^{16} + 0.000195779\,b^{17} + 0.000665182\,a\,b^{17} + 0.00114961\,a^2\,b^{17} +$$
$$0.00131434\,a^3\,b^{17} + 0.00110612\,a^4\,b^{17} + 0.000719411\,a^5\,b^{17} + 0.000364477\,a^6\,b^{17} +$$
$$0.000135235\,a^7\,b^{17} + 0.0000240752\,a^8\,b^{17} - 0.0000141338\,a^9\,b^{17} - 0.0000192837\,a^{10}\,b^{17} -$$
$$0.0000141958\,a^{11}\,b^{17} - 8.34463\times10^{-6}\,a^{12}\,b^{17} - 4.29026\times10^{-6}\,a^{13}\,b^{17} - 2.00121\times10^{-6}\,a^{14}\,b^{17} -$$
$$0.0000136855\,b^{18} - 0.0000506047\,a\,b^{18} - 0.000102194\,a^2\,b^{18} - 0.000143926\,a^3\,b^{18} -$$
$$0.000152562\,a^4\,b^{18} - 0.000127862\,a^5\,b^{18} - 0.0000880433\,a^6\,b^{18} - 0.0000512701\,a^7\,b^{18} -$$

$$0.0000257489\,a^8 b^{18} - 0.0000112708\,a^9 b^{18} - 4.3063\times10^{-6}\,a^{10} b^{18} - 1.42465\times10^{-6}\,a^{11} b^{18} +$$

$$3.47657\times10^{-6}\,a^2 b^{19} + 7.12902\times10^{-6}\,a^3 b^{19} + 9.6854\times10^{-6}\,a^4 b^{19} + 0.0000100326\,a^5 b^{19} +$$

$$8.41548\times10^{-6}\,a^6 b^{19} + 5.91832\times10^{-6}\,a^7 b^{19} + 3.58107\times10^{-6}\,a^8 b^{19} + 1.90788\times10^{-6}\,a^9 b^{19} -$$

$$1.12884\times10^{-6}\,a^2 b^{20} - 1.40938\times10^{-6}\,a^3 b^{20} - 1.39455\times10^{-6}\,a^4 b^{20} - 1.12936\times10^{-6}\,a^5 b^{20} = 0.$$